**Министерство образования и науки Российской Федерации**
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«Владимирский государственный университет имени
Александра Григорьевича и Николая Григорьевича Столетовых»(ВлГУ)**

# МЕТОДИЧЕСКИЕ УКАЗАНИЯ

(вспомогательный теоретический материал)
по дисциплине
"Современные методы кодирования информации"

**Направление подготовки:** 11.04.01 «Радиотехника»

**Квалификация (степень) выпускника:** магистр

**Форма обучения:** очная

Составитель: П.А. Полушин

**Владимир, 2018**

## 1. Обзор основных методов кодирования

Код представляет собой определенную форму представления сообщения, которая может не зависеть от физической сути сигналов, хотя практически она обычно связана с их физическими параметрами. Многие сообщения исходно обладают внутренними корреляционными связями, позволяющими устранять часть возможных ошибок, что, например, делает понятной даже сильно зашумленную речь. Однако в общем случае символы исходного информационного сигнала следует полагать взаимно независимыми, т.к. такая последовательность несет наибольший объем информации. Тем более, если исходные информационные связи выражены слабо, или неизвестны, их затруднительно использовать для повышения помехоустойчивости. В этом случае с помощью кодирования различными методами вводят искусственные связи за счет увеличения числа символов. Степень возникающей избыточности определяет исправляющие свойства кода. (Известны также коды без избыточности, которые используют заранее известный ограниченный объем предаваемых информационных сообщений, однако они, как правило, узкоспециализированы и имеют ограниченное применение).

В настоящее время известно большое количество различных кодов. Основные распространенные виды условно представлены на схеме, приведенной на рисунке 1.1. Важным для диагностики различием выступает разделение на систематические и несистематические коды. В систематических кодах можно в кодовой последовательности выделить исходную информационную последовательность символов. Она может быть непрерывной, либо разделенной на фрагменты. Избыточные новые символы по определенному принципу просто добавляются к исходной последовательности.

В несистематических кодах формируемая в кодере последовательность не содержит неизмененную исходную последовательность. Все символы кодовой последовательности получаются исходя из совокупности исходных символов с использованием определенных функций или правил, определяющих вид и параметры кода. В двоичных кодах каждый символ содержит один информационный бит, символы недвоичных кодов содержат несколько бит. Различие фактически состоит лишь в усложнении аппаратуры кодирования.

Линейные коды отличаются от нелинейных замкнутостью получаемого множества кодовых слов относительно оператора, с помощью которого реализуется кодирование. Оператор в этом случае является линейным, а кодовые слова можно описать в виде векторов некоторого пространства. Линейность кода упрощает его реализацию и при большой длине кодовых слов практически применяются только линейные коды. Линейные коды образуют

обширные классы. Однако нелинейные коды в среднем обладают лучшими характеристиками.


Рисунок 1.1.

Различные коды удобно разделять на относительно простые (немодифицированые) и на модифицированные. Работа посвящена диагностике кодов первого вида, т.к. эти коды выступают как некоторая основа, на базе которой строятся коды второго вида.

Рассматриваемые коды в зависимости от основного правила их получения делятся на сверточные и блоковые. Несмотря на их существенные различия, после определенной модификации каждой группе можно придать некоторые свойства другой группы. Подробнее рассмотрим принципы формирования и свойства каждой из этих групп. Поскольку для целей диагностики имеют основное значение операции процедуры кодирования, то основное внимание будет уделено именно им.

## 2. Сверточные коды

Сверточные коды в настоящее время нашли широкое применение. Они иногда называются непрерывными кодами, потому что используют непрерывную (последовательную) обработку символов. Кроме того они являются линейными кодами. Кодер обладает памятью в том смысле, что каждый выходной кодовый символ зависит не только от текущего входного информационного символа, но и от нескольких предыдущих входных символов.

Избыточность кода, определяется соотношением количества кодовых символов, требующихся для передачи определенного числа информационных символов. Если для передачи $k$ информационных символов требуется $n$ кодовых символов, то кодовая скорость равна $R=k/n$. Обобщенная структурная схема сверточного кодера приведена на рисунке 2.1.

На его вход поступает последовательность информационных символов $\mathbf{m}=m_1,m_2,\ldots$ На выходе образуется последовательность кодированных символов $\mathbf{u}=u_1,u_2,\ldots$ Кодер состоит из $k$ сдвиговых регистров, содержащих по $K$–1 ячеек. Входной коммутатор (Комм.1) распределяет символы входной последовательности по последовательным входам регистров. При каждом такте содержимое регистров сдвигается в соседнюю ячейку. Сигналы с параллельных выходов регистров подаются на входы $n$ многовходовых сумматоров $n>k$. В сумматорах над входными сигналами производится логическая операция сложения по модулю 2 («исключающее или»). Входы каждого сумматора подключены к своему определенному набору ячеек сдвиговых регистров. Эти наборы различаются у всех сумматоров и определяют структуру конкретного используемого кодера.

Выходная кодовая последовательность образуется последовательным поочередным подключением с помощью коммутатора (Комм.2) выходных сигналов сумматоров на общий выход кодера. Таким образом, при поступлении $k$ исходных информационных символов образуется $n$ кодовых символов. Варьируя параметры $k$ и $n$, можно регулировать кодовую скорость $R$. Однако чаще используются коды со скоростью $1/n$, а скорость регулируется применением перфорации (выкалывания).

В формировании каждого кодового символа в данный момент времени участвует входной символ и $K$–1 предыдущих входных символов. Таким образом кодовое ограничение $K$ определяет память кодовой последовательности. Кодер, приведенный на рисунке 2.1, представляет собой систему с конечным откликом.

Если при $k$=1 пропускать через кодер исходную последовательность символов, содержащую только одну единицу, а остальные – нули, то формируемая кодовая последовательность будет представлять собой набор импульсных откликов $g_1 \div g_n$ каждого из $n$ сумматоров. Наборы коэффициентов $g_1 \div g_n$ описываются в виде векторов **g**. Размеры векторов составляют максимум $K$ двоичных элементов и полностью определяют структуру кодера. Каждый разряд соответствующего двоичного кода соответствует одному отводу регистра, и единица в нем устанавливает факт наличия связи данного разряда регистра с сумматором. Двоичные последовательности **g** (кодовых генераторов) обычно записывают в десятичной форме или разделяют на группы по три символа и записывают в восьмеричной форме. Также используется запись **g** в виде полинома (порождающего полинома).

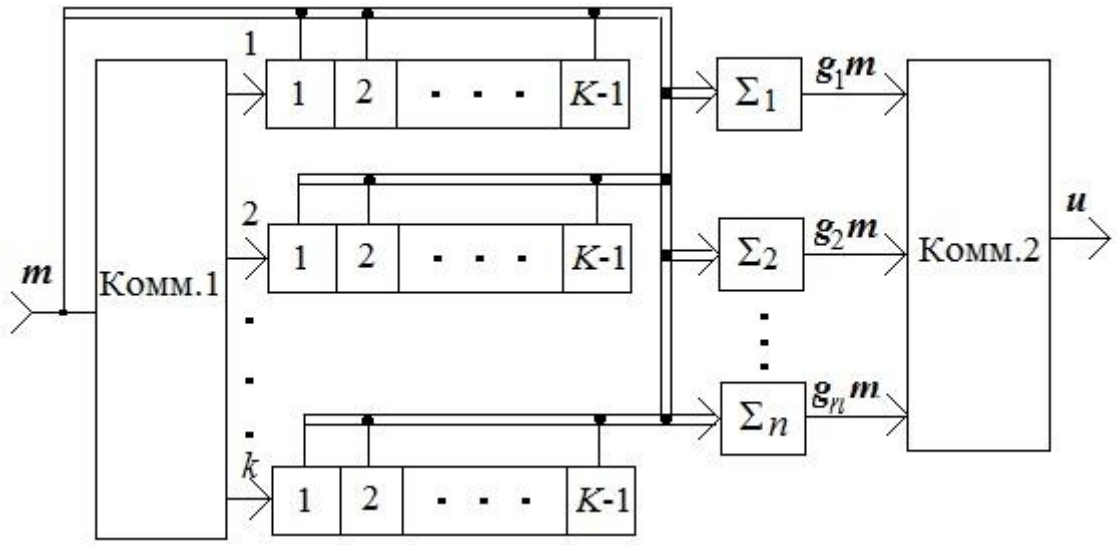



Рисунок 2.1.

Поскольку выходной сигнал каждого сумматора представляет собой свертку соответствующего порождающего полинома и фрагмента входной информационной последовательности **m**, состоящей и текущего входного символа и $K$–1 предыдущих входных символов, то формируемые группы по $n$ кодовых символов, соответствующих одному входному символу, можно записать в векторной форме: $\mathbf{u}_0=u_1,\ldots,u_n$, где $u_1=\mathbf{g}_1\mathbf{m}$, $u_2=\mathbf{g}_2\mathbf{m},\ldots,u_n=\mathbf{g}_n\mathbf{m}$ (произведением обозначена свертка векторов). Формируемую кодовую

последовательность также можно получить с помощью некоторой матрицы **G** (порождающей матрицы), имеющей ленточную структуру.

Исправляющие свойства сверточных кодов с разными наборами порождающих полиномов различаются. Для удобного графического исследования работы кодера и декодера эффективно использование решетчатых диаграмм. После достижения глубины анализа, равной $K$ шагов, структура диаграммы становится постоянной. Решетчатая диаграмма используется при декодировании. Правильно декодированная кодовая последовательность должна соответствовать на каждом шаге только одному из нескольких разрешенных путей по решетке. Суммарное количество отличий принятой последовательности от разрешенных вариантов, измеренное в определенной метрике, служит указанием для восстановления передаваемого информационного сообщения и освобождения его от ошибок. Для каждого набора исходных данных (кодовой скорости, кодового ограничения) известны наиболее эффективные в смысле исправления ошибок коды, хотя могут использоваться и коды с близкой структурой, несколько уступающие им по эффективности.

Структура систематических сверточных кодеров незначительно отличается от структуры несистематических кодеров, представленной на рисунке 2.1. Отличия заключаются в том, что один из входов выходного коммутатора подключен не к выходу одного из сумматоров по модулю 2, а непосредственно ко входу кодера. В результате одна из компонент формируемого кодовой последовательности совпадает со входной информационной последовательностью, хотя ее символы следуют не один за другим, а разделены n–1 другими кодовыми символами. Систематические сверточные коды используются редко, т.к. их характеристики в среднем хуже, чем у соответствующих несистематических кодов.

Кроме требования высокой эффективности исправления ошибок на вид полиномиальных генераторов накладываются и другие ограничения. Кодовые генераторы удобно представлять в виде полиномов вида:

$$\mathbf{g}(X)=a_0+a_1X+a_2X^2+\ldots+a_nX^n,$$

где переменная $X$ обозначает сдвиг по времени на один такт, $X^2$ – сдвиг по времени на два такта, и т.д.; коэффициенты $a_0 \div a_n$ могут принимать нулевое или единичное значение. Важное ограничение на вид полиномов связано с возможностью распространения катастрофических ошибок при декодировании, когда конечное число ошибок в кодовых словах вызывает бесконечное число ошибок в декодированных данных.

Возможность для появления катастрофических ошибок появляется, если при разложении используемых порождающих полиномов на более простые полиномы-множители у любых двух из них в разложении будет присутствовать хотя бы один одинаковый простой полином-множитель.

Известны также рекурсивные сверточные коды, которые обычно бывают систематическими. Они отличаются тем, что характеризуются бесконечным импульсным откликом и сумматоры могут находиться не только на выходах, но и на входах регистров. Несистематический кодер и систематический сверточный кодер имеют одно и т же множество кодовых последовательностей, но с другим соответствие между информационными и кодовыми словами.

## 3. Блоковые коды

Первоначально рассмотрим двоичные блоковые коды. Виды известных блоковых кодов характеризуются исключительным разнообразием, связанным с особенностями кодирования и декодирования, сложностью реализации, быстродействием и помехоустойчивостью. Однако для целей диагностики большинство методов их получения можно объединить в две связанные между собой группы. В одной из них используются порождающие матрицы, в другой группе – порождающие полиномы.

Любой блоковый код основан на том, что исходная информационная последовательность символов разбивается на группы, как правило, одинаковой длины $k$. По определенному правилу на основе значений информационных символов в группе вычисляется последовательность из $b$ проверочных символов и добавляется к информационной последовательности, формируя выходной кодовый блок длиной $n$.

Если длина информационной части блока невелика, то наиболее простым методом кодирования является использование таблицы соответствия. Таблица соответствия содержит все варианты информационных последовательностей и соответствующие им варианты кодовых блоков. При кодировании на основе каждой новой информационной части для передачи просто выбирается нужный кодовый блок. Однако размер таблицы пропорционален $2^k$, поэтому при большой длине информационных частей величина таблицы и время кодирования могут стать недопустимо большими.

В этом случае задачу можно упростить, если не хранить в памяти кодовые блоки, соответствующие поступающим информационным группам, и не отыскивать их в таблице, а каждый раз генерировать вновь. Пусть $\mathbf{m}$ – вектор, содержащий $k$ двоичных элементов и составленный из группы исходных информационных символов. Тогда, если имеется

произвольный базис из $k$ линейно независимых двоичных векторов, тогда любой информационный вектор можно записать, как линейную комбинацию некоторых векторов этого базиса. (При составлении линейной комбинации под умножением понимается операция «и», под сложением – операция «исключающее или»). Таким образом, если имеется набор $k$ двоичных линейно независимых векторов $\mathbf{V}_1$, $\mathbf{V}_2$,…, $\mathbf{V}_k$ , каждый из которых состоит из $n$ элементов, то вектор $\mathbf{u}$, описывающий любой кодовый блок, можно записать, как: $\mathbf{u}=m_1\mathbf{V}_1+m_2\mathbf{V}_2+\ldots+m_k\mathbf{V}_k$.

Порождающая матрица $\mathbf{G}$ имеет размер $k\times n$ и представляет собой набор векторов $\mathbf{V}_1 \div \mathbf{V}_k$, как строк:

$$\mathbf{G} = \begin{Vmatrix} \mathbf{V}_1 \\ \mathbf{V}_2 \\ \vdots \\ \mathbf{V}_k \end{Vmatrix} = \begin{Vmatrix} v_{1,1}, v_{1,2}, \ldots, v_{1,n} \\ v_{2,1}, v_{2,2}, \ldots, v_{2,n} \\ \ldots\ldots\ldots \\ v_{k,1}, v_{k,2}, \ldots, v_{k,n} \end{Vmatrix},$$

где $v_{i,j}$ – элементы с индексом $j$ вектора $\mathbf{V}_i$.

Генерация кодового слова в матричной форме описывается уравнением $\mathbf{u}=\mathbf{mG}$. При этом при кодировании в памяти необходимо сохранять только $k$ векторов $\mathbf{V}_1 \div \mathbf{V}_k$ , а не $2^k$, как при использовании таблицы соответствия.

Матрицу $\mathbf{G}$ можно представить, как состоящую из двух блоков, первый блок $\mathbf{G}_1$ размером $k\times k$, второй блок $\mathbf{G}_2$ размером $k\times(n–k)$, т.е.: $\mathbf{G} = \|\mathbf{G}_1 \vdots \mathbf{G}_2\|$ . Матрица $\mathbf{G}_1$ определяет вид информационной части кодового слова, матрица $\mathbf{G}_2$ определяет вид проверочной части кодового слова. При использовании несистематического кодирования вид матрицы $\mathbf{G}_1$ может быть произвольным (при сохранении условия линейной независимости строк). При использовании систематического кодирования $\mathbf{G}_1$ является единичной матрицей, $\mathbf{G}_1=\mathbf{E}$. Несистематическое кодирование обеспечивает такие же характеристики помехоустойчивости, как и систематическое, однако при использовании систематического кодирования процесс кодирования существенно упрощается и ускоряется ([ ]), поэтому практически используется систематическое кодирование. К тому же порождающая матрица для несистематического кодирования легко преобразуется в матрицу для систематического кодирования умножением на обратную матрицу $\mathbf{G}_1^{-1}$.

После переобозначения части матрицы, отвечающей за формирование проверочной части кодового слова, $\mathbf{G}_2=\mathbf{P}$, порождающая матрица приобретает вид: $\mathbf{G} = \|\mathbf{E} \vdots \mathbf{P}\|$. При подробном рассмотрении структуры генерируемого кодового слова видно, что каждый

проверочный бит имеет свое происхождение и является «индивидуальной» комбинацией каких-либо информационных битов. При этом очевидно, что по сравнению с контролем четности методом дублирования разряда или применением одного бита четности, применяемый метод обеспечивает более широкие возможности для исправления возникающих при передаче ошибок.

*Циклические коды* представляют класс блоковых кодов, кодирование и декодирование которых основано на использовании полиномиальных представлений, более простых, чем матричные процедуры. Широкое распространение этот класс кодов получил в связи с удобством практической реализации на основе достаточно простой современной цифровой элементной базы. Линейный код называется циклическим, если при любом циклическом сдвиге некоторого кодового слова получается другое кодовое слово, которое может быть получено тем же кодером, но из другой входной информационной группы.

Значения символов кодового слова можно рассматривать, как коэффициенты полинома: $\mathbf{u}(X)=u_0+u_1X+u_2X^2+\ldots+u_{n-1}X^{n-1}$. Наличие или отсутствие каких-либо членов в полиноме означает наличие нулей в соответствующих разрядах кодового слова. Если $u_{n-1}\neq 0$, то порядок полинома равен $n-1$.

Циклический код создается с помощью полиномиального генератора $\mathbf{g}(X)$. Для заданного циклического кода $(n, k)$ вид полиномиального генератора является единственным:

$$\mathbf{g}(X)=g_0+g_1X+g_2X^2+\ldots+g_bX^b .$$

Предполагается, что коэффициенты $g_0$ и $g_b$ – ненулевые, $b=n-k$. Каждый полином, описывающий какое-либо кодовое слово, имеет вид $\mathbf{u}(X)=\mathbf{m}(X)\mathbf{g}(X)$, т.е. кодовое последовательность двоичных символов действительно является кодовым словом, сформированным данным кодером, если оно делится без остатка на $\mathbf{g}(X)$. Полиномиальный генератор является одним простым множителем или произведением нескольких простых множителей полинома $X^n+1$. Использование циклического кодирования в описанном виде формирует несистематические коды, т.е. в кодовом слове $\mathbf{u}(X)$ нет фрагмента, все символы которого совпадали бы с вектором $\mathbf{m}(X)=m_0+ m_1X+m_2X^2+m_3X^3+\ldots+m_{k-1}X^{k-1}$ .

Однако чаще используется систематическая форма кодов, упрощающая процедуры обработки. При систематическом кодировании вектор $\mathbf{m}(X)$ совпадает с фрагментом выходного кодового слова $\mathbf{u}(X)$ и используется как его часть. Для этого символы информационного сообщения первоначально сдвигаются в сторону бóльших степеней $X$ на $n-k$ позиций с помощью умножения на $X^{n-k}$. При этом формируется полином:

$$X^{n-k}\,\mathbf{m}(X)= m_0X^{n-k}+ m_1X^{n-k+1}+m_2X^{n-k+2}+m_3X^{n-k+3}+\ldots+m_{k-1}X^{n-1}.$$

Далее он делится на выбранный порождающий полином $\mathbf{g}(X)$. Результат деления можно записать в виде:

$$X^{n-k}\,\mathbf{m}(X)=\mathbf{q}(X)\mathbf{m}(X)+\mathbf{p}(X), \qquad (3.1)$$

где $\mathbf{q}(X)$ – частное от деления; $\mathbf{p}(X)$– остаток от деления, равный:

$$\mathbf{p}(X)=p0+p1X+p2X^2+\ldots+p^{n-k-1}\,X^{n-k-1}.$$

Остаток от деления суммируется по модулю 2 с обеими частями уравнения (3.1), в результате получается полином:

$$\mathbf{u}(X)=X^{n-k}\,\mathbf{m}(X)+\mathbf{p}(X)=\mathbf{q}(X)\mathbf{m}(X).$$

Полученный таким образом полином $\mathbf{u}(X)$ является кодовым словом, поскольку делится на $\mathbf{g}(X)$, но его фрагмент из $k$ символов, стоящий в области больших степеней, теперь уже совпадает с информационным вектором $\mathbf{m}(X)$, а фрагмент, стоящий в области $n–k$ меньших степеней, является проверочной частью кодового слова, т.е.:

$$\mathbf{u}(X)=p_0+p_1X+p_2X^2+\ldots+p_{n-k-1}\,X^{n-k-1}+m_0X^{n-k}+m_1X^{n-k+1}+\ldots+m_{k-1}\,X^{n-1}.$$

Процедура кодирования иллюстрируется структурной схемой, приведенной на рисунке 3.1. Схема состоит из последовательно включенных элементов памяти $Э_1 \div Э_{n-k-1}$. Фактически она представляет собой регистр сдвига с обратными связями. В момент прихода тактового импульса каждый элемент памяти запоминает значение символа на своем входе и хранит его до поступления следующего тактового импульса. Между ними находятся сумматоры по модулю 2. На их входы поступают выходные сигналы с элементов памяти и с элементов $g_1 \div g_{n-k-1}$. Если коэффициент $g_1 \div g_{n-k-1}$. равен единице, то выходной сигнал ключа (Кл.) подается на соответствующий сумматор, если равен нулю, то суммирование не производится.

Схема работает следующим образом. Моменты появления тактовых импульсов совпадают с появлением информационных символов. При первых символах последовательности $\mathbf{m}(X)$ ключ закрыт, а коммутатор (Комм.) подключает на выход сигнал со своего первого входа, т.е. непосредственно исходную информационную последовательность. После передачи $k$-го информационного символа ключ открывается, а

коммутатор на свой выход подключает сигнал со второго входа. В следующих $n–k$ тактах происходит формирование проверочных символов. После окончания $n$ тактов обработки в выходном регистре (Р) оказываются записаны все сформированные символы кодового слова, которые далее поступают в передатчик. Для повышения скорости работы кодера иногда используют комбинацию табличного метода и регистра с обратными связями.

Некоторые виды блоковых кодов в силу различных причин используются особенно часто. Рассмотрим некоторые из них.

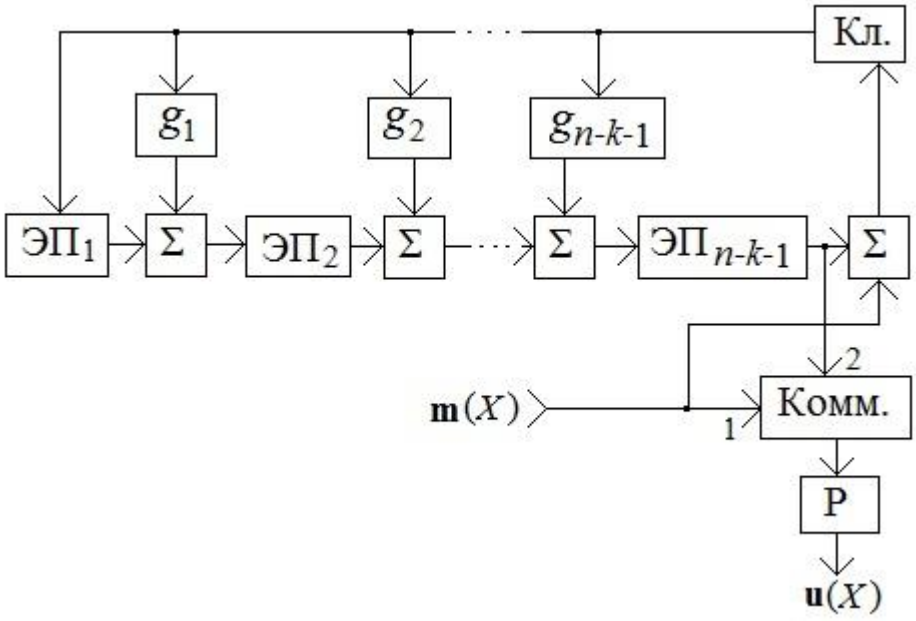



Рисунок 3.1.

*Коды Хемминга.* Это достаточно простой вид блоковых кодов, параметры которых выбираются из следующих условий: $n=2^m–1$; $k=2^m–1–m$, где m=2, 3, 4, …

Они способны исправлять все одиночные ошибки в кодовом слове. Декодирование кодов Хемминга производится достаточно просто. Коды Хемминга несмотря на относительно небольшую эффективность по сравнению с другими кодами, достаточно привлекательны тем, что они требуют при заданной блине блока минимальную избыточность для исправления одной ошибки. Кодирование производится обычно с использованием соответствующей матрицы.

*Код Голея.* Это линейный блоковый код с параметрами (n,k)=(23,12). Он допускает исправление до трех ошибок в кодовых словах длиной 23 символа. Из-за относительно небольшой его длины кодирование и декодирование двоичного кода Голея может быть выполнено даже с использованием соответствующих таблиц. В этом случае таблица содержит список из $2^{12}$=4096 кодовых слов, пронумерованных непосредственно двоичной

совокупностью соответствующих информационных частей блока. Код Голея имеет циклическую природу и может быть получен и помощью циклического кодирования. Его порождающий полином равен $\mathbf{g}(X)=1+X^2+X^4+X^5+X^6+X^{10}+X^{11}$.

Степень кодирования кода равна 11/23, что не всегда удобно в практическом осуществлении, поэтому часто применяется близкий к нему расширенный код Голея с параметрами (24,12), получаемый добавлением к кодовому слову еще одного проверочного символа. Его кодовая скорость равна ½ и практическая реализация проще. Код может исправлять в кодовом слове кроме всех трехсимвольных ошибок также некоторое число четырехсимвольных ошибок и является существенно более мощным, чем код Хемминга.

*Коды Рида-Малера* также представляют собой линейные блоковые коды. Для кодирования используются соответствующие матричные процедуры. Используются коды с разными наборами параметров. Привлекательность кодов основана на достаточно простой процедуре декодирования, базирующейся на мажоритарной логике, при которой решение о значении символов принимается по большинству результатов, полученных из соответствующих проверочных уравнений.

*Коды БЧХ* (Боуза-Чоудхури-Хоквенгема). Эти коды можно считать обобщением кодов Хэмминга, позволяющим исправлять несколько ошибок в блоке. Они позволяют относительно просто менять параметры кодирования, варьируя длину блока, кодовую скорость и возможности коррекции ошибок. Порождающий полином всегда имеет длину $n–k$. Коды бывают как двоичные, так и недвоичные. В наиболее часто используемых кодах применяются блоки длиной $n=2^m–1$, m=2,3,4,… . Известны таблицы порождающих полиномов наиболее эффективных кодов. Коды задаются корнями порождающего полинома. Иногда эффективность кодов с большой избыточностью хуже, чем кодов с меньшей избыточностью. Наибольшая эффективность достигается в зависимости от кодовой скорости при фиксированном $n$ находится ориентировочно в интервале значений $R$ от 1/3 и ¾ [ ].

Коды Рида-Малера, БЧХ и Рида-Соломона относятся к полиномиальным кодам, которые отличаются тем, что на кони их порождающих полиномов накладываются дополнительные условия.

## 4. Недвоичные и нелинейные коды

*Недвоичные коды.* Недвоичные коды требуют более сложной обработки, чем двоичные, однако реализуют бóльшую эффективность. В качестве символов недвоичных кодов обычно рассматриваются группы бит. Наиболее часто используются коды Рида-Соломона, являющиеся недвоичными БЧХ кодами.

Коды Рида-Соломона являются множеством кодовых слов, компоненты которых равны значениям некоторых определенных полиномов. Обработка значений символов производится в соответствии с правилами вычислений в полях Галуа. Для их описания применяются специальные символы α, обозначающие дополнительные однозначные элементы алгебры конечных полей используемой размерности. Коды позволяют исправлять не только несколько одиночных ошибок, но и пакеты ошибок. Обычно используется систематическая форма кодов. Кодирование при этом осуществляется схемой, сходной со схемой на рисунке 3.1 для кодирования двоичных кодов.

Элементы памяти запоминают не двоичный символ, содержащий один бит информации, а символ, несущий несколько информационных бит. Вместо двоичных коэффициентов $g_i$ используются коэффициенты, образованные из степеней α, которые в данном случае также представляют несколько бит информации. Сложение, как и умножение, производится по правилам операций в конечных полях.

Коды Рида-Соломона позволяют исправить любой набор из $[(n–k)/2]$ ошибок в слове или любой набор $n–k$ стираний (т.е. определить место в блоке, в котором произошла ошибка без последующего ее исправления). Также имеется возможность одновременной коррекции определенного количества ошибок и стираний. Коды обладают существенными преимуществами перед двоичными кодами при воздействии коротких импульсных помех. Поскольку исправляется не бит, а символ с несколькими битами, то длина помехи может составлять и несколько бит, если она не выходит за длительность символа.

*Нелинейные коды.* Известны различные виды нелинейных кодов (Коды с контрольной суммой, инверсные, коды на основе перестановок, с повторением и без повторения символов, и т.д.). Построение нелинейных кодов сложнее, чем линейных, границы возможных значений параметров обычно связаны сложным образом и более узкие, чем у линейных кодов. В настоящее время коды находят ограниченное применение.

В следующем параграфе будут рассмотрены более сложные коды, основой для которых являются, как правило, сверточные и блоковые коды.

### 5. Виды сложных кодов. Модифицированные, укороченные и расширенные коды

Под сложными кодами часто понимаются коды, которые получаются различными операциями над простыми кодами, при этом под простыми кодами можно понимать полученные с помощью методов сверточного и блокового кодирования. Здесь будут рассмотрены коды, модифицированные добавлением или удалением символов; комбинации, получаемые совместным использованием кодов; коды, предназначенные для декодирования,

проводимого повторением нескольких последовательных процедур (итеративно декодируемые коды); коды, использующие особенности модуляции сигналов (сигнально-кодовые конструкции).

*Модифицированные коды.* Они строятся на основе некоторого исходного кода, к которому либо добавляют дополнительные символы (расширенный код), либо сокращают часть информационных символов без изменения расстояния между кодовыми символами (укороченный код), либо выбрасывают часть символов (перфорация или выкалывание). Таким образом, достигается определенная гибкость в получении нужных параметров передачи информации. Например, одной из ситуаций, когда эффективно укорочение, является потребность применить код Хэмминга, но при этом требуется число кодовых символов не равное $2^m$, $m$ – целое.

*Укорочение кодов*. Укорочение производится отбрасыванием определенного числа $s$ позиций в информационной части кодового слова ($s$ – глубина укорочения). В слове на передачу информации теперь отводится только $k–s$ символов, но длина проверочной части $n–k$ остается прежней. Укороченное кодовое слово формируется посредством того, что в некоторых фиксированных позициях информационной части исходного (неукороченного) кодового слова устанавливаются нули. Остальные позиции могут принимать произвольные значения в зависимости от информации, передаваемой в этом кодовом слове. Положение нулевых позиций принципиального значения не имеет, однако для практического удобства кодирования и декодирования обычно выбирается $s$ старших позиций. Таким образом, информационная часть кодового слова приобретает вид:

$$\mathbf{m}(X)=m_0+ m_1X+m_2X^2+m_3X^3+\ldots+m_{k-1}X^{k-1-s}.$$

Далее оно также систематическим кодером сдвигается на $n–k$ разрядов, делится на порождающий полином и остаток от деления записывается в младшие пустые разряды. Образуется укороченный линейный код, степень которого не превышает $n–s–1$. В общем случае такой код не остается циклическим.

Если для кодирования используется порождающая матрица $\mathbf{G} = \|\mathbf{E} \vdots \mathbf{P}\|$, то она также должна быть модифицирована. Для этого из нее удаляются s выбранных столбцов и одновременно s строк, соответствующих ненулевым элементам удаленных столбцов.

Пусть исходную матрицу можно записать в виде:

$$\mathbf{G} = \begin{Vmatrix} 1,0,0,.....,0, g_{1,k+1}, g_{1,k+2},....., g_{1,n} \\ 0,1,0,.....,0, g_{2,k+1}, g_{2,k+2},....., g_{2,n} \\ ..... \\ 0,0,0,.....,1, g_{k,k+1}, g_{k,k+2},..... g_{k,n} \end{Vmatrix} = \begin{Vmatrix} 1,0,0,.....,0, p_{1,k+1}, p_{1,k+2},....., p_{1,n} \\ 0,1,0,.....,0, p_{2,k+1}, p_{2,k+2},....., p_{2,n} \\ ..... \\ 0,0,0,.....,1, p_{k,k+1}, p_{k,k+2},..... p_{k,n} \end{Vmatrix}.$$

Если при укорочении удаляется $s$ первых столбцов (и соответственно, $s$ первых строк), то размер новой порождающей матрицы становится равным $(k–s)\times(n–s)$. Единичная информационная матрица становится размером $(k–s)\times(k–s)$. Элементы новой укороченной проверочной матрицы: $g_{i,j} = p_{i–s,j–s}$. Операция укорочения уменьшает длину информационной части, в то время как число проверочных символов остается прежним. В результате может быть исправлено большее число комбинаций ошибок, и помехоустойчивость передачи информации возрастает. Таким образом, укороченные коды более эффективны, чем исходные неукороченные коды.

Важное свойство укороченных кодов состоит в том, что для работы с ними могут быть использованы те же самые кодеры и декодеры, что и для неукороченных кодов. Однако для них обычно не выполняется правило циклического сдвига. При практической реализации слова обычно дополняются нулями в откинутых старших разрядах и используются те же самые алгоритмы кодирования и декодирования. А при передаче сообщений включенные нули не передаются.

*Расширение кодов.* Расширение кодов означает добавление в кодовое слово проверочных символов при сохранении в нем количества информационных символов. В результате получается код $(n+s,k)$. Расширенная порождающая матрица может быть получена из исходной матрицы добавлением к ней соответствующих $s$ столбцов.

Ее размер становится равным $k\times(n+s)$. Если вводимые новые элементы обозначить через $q_{i,j}$, то расширенная матрица приобретает вид:

$$\mathbf{G} = \begin{Vmatrix} 1,0,0,.....,0, g_{1,k+1}, g_{1,k+2},....., g_{1,n}, q_{1,n+1},..., q_{1,n+s} \\ 0,1,0,.....,0, g_{2,k+1}, g_{2,k+2},....., g_{2,n}, q_{2,n+1},..., q_{2,n+s} \\ ..... \\ 0,0,0,.....,1, g_{k,k+1}, g_{k,k+2},..... g_{k,n}, q_{k,n+1},..., q_{k,n+s} \end{Vmatrix}.$$

Расширение кодов также увеличивает помехоустойчивость и изменяет кодовую скорость.

Иногда рассматривается объединенная операция расширения, модифицирующая исходный код и состоящая в добавлении в порождающую матрицу и новых строк, и новых

столбцов. Ее размер становится равным $(k+s)\times(n+s)$. При этом ее часть, имеющая вид единичной матрицы, занимает не одну из половин, а меньшую ее часть. Матрица приобретает вид:

$$\mathbf{G} = \begin{Vmatrix} 1,0,0,\ldots..,0, g_{1,k+1}, g_{1,k+2},\ldots\ldots\ldots, g_{1,n}, q_{1,n+1},\ldots, q_{1,n+s} \\ 0,1,0,\ldots..,0, g_{2,k+1}, g_{2,k+2},\ldots\ldots, g_{2,n}, q_{2,n+1},\ldots, q_{2,n+s} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ 0,0,0,\ldots..,1, g_{k,k+1}, g_{k,k+2},\ldots\ldots g_{k,n}, q_{k,n+1},\ldots, q_{k,n+s} \\ q_{k+1,1}, q_{k+1,2}, q_{k+1,3},\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots, q_{k+1,n+s} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ q_{k+s,1}, q_{k+s,2}, q_{k+s,3},\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots, q_{k+s,n+s} \end{Vmatrix}.$$

Кроме рассмотренных вариантов модификации известны также варианты, когда к множеству всех возможных кодовых слов добавляются новые кодовые слова (операция пополнения) или из этого множества удаляются некоторые кодовые слова (операция выбрасывания). Для осуществления операции пополнения при сохранении линейности кода в порождающую матрицу добавляются новые линейно независимые строки.

Операция выбрасывания в общем случае приводит к нелинейному коду. Для сохранения линейности возможно удалить часть строк исходной порождающей матрицы. Для систематических кодов операции выбрасывания и укорочения совпадают.

## 6. Перфорированные коды

Перфорация сверточных кодов применяется в основном для сверточных кодов и состоит в том, что из процесса передачи в канал удаляются некоторые символы с выхода кодера. Поскольку при этом структура собственно кодера не меняется, то и количество информационных символов не меняется. В результате формируется перфорированный код с более высокой кодовой скоростью.

Правило удаления символов задается, как правило, матрицей (маской) перфорации. Матрица перфорации $P$ задает правило удаления выходных символов. Обычно процесс перфорации является периодическим и сходный (неперфорированный) код имеет скорость $R=1/n$, $n$ – целое. В этом случае количество $N_p$ кодовых символов, через которое процесс повторяется, должно быть кратно $n$.

Элементы матрицы $P$ равны нулю или единице. Ноль указывает, что соответствующий двоичный символ будет удален, единица – что он будет оставлен. Размеры

матрицы $P$ равны $n \times N_p$. Если она содержит $q$ нулей, то кодовая скорость после перфорации станет равной $R=N_p/(nN_p-q)$.

Формирование перфорированного кода производится следующим образом. После прихода на исходный кодер $N_p$ информационных символов он вырабатывает $N_p$ групп кодовых символов, каждая из которых соответствует одному из исходных символов. Эти группы можно записать в виде векторов размера $n$. Далее символы каждого из векторов соотносятся с соответствующим столбцом маски перфорации. Если элемент маски равен нулю, то символ с таким номером из вектора удаляется. (После этого размеры векторов становятся неравными.) Всего будет удалено $q$ символов. Далее из оставшихся элементов всех векторов последовательно составляется перфорированная последовательность длиной $nN_p-q$. После этого аналогично производится следующий период обработки.

При записи структуры перфорированного кода правило получения каждого кодового символа как обычно записывается в форме восьмеричного кода, а место перфорированных символов обозначается буквой $X$. Перфорация используется достаточно широко, поскольку для различных вариантов перфорированного кода, полученных из одного и того же исходного кода, можно использовать один и тот же декодер, предназначенный для исходного кода.

Перфорация применяется также и к линейным блоковым кодам. Она состоит в удалении из кодового слова $p$ проверочных символов. Код при этом остается линейным блоковым, количество информационных символов $k$ в кодовом слове не изменяется, а общее количество кодовых символов уменьшается до $n-p$. Избыточность уменьшается, а скорость кода возрастает.

## 7. Комбинированные и каскадные коды

*Комбинированные коды.* Известны различные способы комбинирования кодов. Рассмотрим метод последовательного соединения (concatenation – конкатенация). Если используются блоковые коды $C_1$ и $C_2$, то их последовательное соединение эквивалентно поочередной передаче кодовых слов, полученных с помощью первого кода и второго кода. В результате последовательного соединения $i=1 \div m$ кодов образуется комбинированный код с параметрами $n_0$ и $k_0$:

$$n_0 = \sum_{i=1}^{m} n_i \ , \quad k_0 = \sum_{i=1}^{m} k_i \ ,$$

где $n_i$ и $k_i$ – параметры $i$-го кода. Если $\mathbf{G}_i$ – исходные порождающие матрицы кодов, составляющих последовательное соединение, то его порождающая матрица $\mathbf{G}_0$ последовательного соединения кодов будет равна:

$$\mathbf{G}_0 = \begin{Vmatrix} \mathbf{G}_1,0,0,....,0 \\ 0,\mathbf{G}_2,0,....,0 \\ ..... \\ o,o,o,.....,\mathbf{G}_m \end{Vmatrix}.$$

Величины $n_i$ и $k_i$ у объединяемых кодов чаще не совпадают, т.к. последовательно соединенные коды используют в случае необходимости реализации разного уровня защиты от ошибок. Если же все $n_i$ и $k_i$ одинаковы, это эквивалентно $m$-кратной повторной передаче одного кодового слова.

Другим типом комбинированных кодов являются *произведения кодов*. В простейшем виде они представляют собой последовательное (повторное) кодирование. При этом выход первого кодера является входом второго кодера, и т.д. (рисунок 7.1.). Несмотря на простоту, метод формирует достаточно помехоустойчивые коды, особенно для кодов с низкой кодовой скоростью. Если используются два кодера, то код первого из них обычно называется внешним кодом, второй – внутренним кодом.

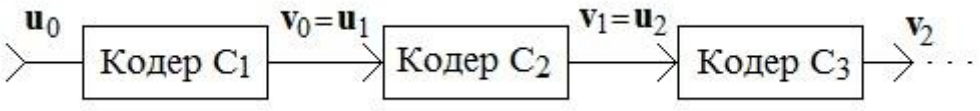



Рисунок 7.1.

Если $\mathbf{G}$ и $\mathbf{H}$ – порождающие матрицы двух кодов с размерами $m \times n$ и $p \times r$, то порождающая $\mathbf{Q}$ матрица произведения этих кодов есть кронекерово произведение этих матриц, т.е.:

$$\mathbf{Q} = \begin{Vmatrix} g_{1,1}\mathbf{H}, g_{1,2}\mathbf{H},...,g_{1,n}\mathbf{H} \\ g_{2,1}\mathbf{H}, g_{2,2}\mathbf{H},...,g_{2,n}\mathbf{H} \\ ..... \\ g_{m,1}\mathbf{H}, g_{m,2}\mathbf{H},...,g_{m,n}\mathbf{H} \end{Vmatrix},$$

где $gij$– элементы матрицы **G**. Матрица **Q** имеет размер $mp \times nr$ . Если параметры исходных кодов равны $(n_1,k_1)$ и $(n_2,k_2)$, то при генерации такого кода кодовые слова длиной $n_1$ символов, уже закодированные внешним кодом, записывают по строкам прямоугольного массива с $n_1$ столбцами, размещая один символ в один элемент массива. Так заполняют $k_2$ строк, а оставшиеся $n_2–k_2$ строк заполняются проверочными символами внутреннего кода. При этом $k_2$ элементов каждого столбца считаются информационными символами внутреннего кода. Общее кодовое слово полученного кода передается из массива по столбцам.

*Каскадные коды.* В каскадных кодах чаще всего используются в качестве внешнего кода недвоичные блоковые коды Рида-Соломона. Символы полученных кодовых слов подвергаются перемежению, в результате чего порядок следования символов изменяется. Далее они подвергаются кодированию с помощью внутреннего кодера, который может быть как блоковым, так и сверточным. Если внешний, и внутренний коды $C_1$ и $C_2$ имеют параметры $(n_1, k_1)$ и $(n_2, k_2)$, то параметры сформированного каскадного кода равны $(n_1n_2, k_1k_2)$ . Когда внутренний код – двоичный линейный блоковый, тогда и полученный каскадный код также является двоичным линейным блоковым.

Были предложены обобщенные каскадные коды, как усложнение каскадных кодов. Они способны исправлять как случайные ошибки, так и случайные пакеты ошибок. Обобщение заключается в том, что вводится разбиение внутреннего кода на подкоды со своей иерархией, а также используется несколько внешних кодов в соответствии с иерархией. При этом если необходимо, то с помощью выбора параметров внутренних и внешних кодов легко получается блоковый или сверточный код с неравной защитой от ошибок. Сообщения, закодированные на верхнем уровне иерархии, имеют усиленную корректирующую способность по сравнению с нижними уровнями.

## 8. Турбокоды

Турбокоды могут быть отнесены как к классу комбинированных составных кодов, так и к итеративно декодируемым кодам, так как несут в себе особенности обоих классов. Они образуются компоновкой нескольких (чаще двух) кодов, создаваемых простыми кодерами (компонентными) и получаемых по-разному из одной и той же информационной последовательности.

Считается, что основной выигрыш при декодировании обусловлен следующим. Обычный декодер и при «жестком», и при «мягком» декодировании выдает «жесткие» двоичные решения. Если же применен каскадный код и декодирование производится в две

ступени, то первый декодер не должен обязательно выдавать «жесткие» решения, т.к. его выходной результат не является окончательным результатом декодирования, а имеет промежуточный характер. Поэтому он тоже может выдавать «мягкие» решения, что значительно повышает помехоустойчивость. Точно также, если общий код скомпонован из двух различных кодов, то при параллельной обработке результат декодирования каждого из кодов поступают в целях коррекции результата декодирования на один из входов декодера другого кода и учитываются им. Причем такой результат выдается с «мягком» виде на основе метрики, построенной использованием отношения правдоподобия.

Достаточно эффективные коды получаются даже при простых используемых компонентных кодерах. Если такие кодеры осуществляют сверточное кодирование, то более эффективны систематические рекурсивные коды. Они формируются кодерами с обратной связью, имеющими бесконечную импульсную характеристику. На рисунке 8.1 приведен пример структурной схемы подобного турбокодера, содержащего два компонентных рекурсивных сверточных кодера, хотя на их количество ограничений нет. Входная информационная последовательность **m** поступает на вход коммутатора (Комм.), а также на первый кодер $C_1$ и через блок чередования (БЧ) на второй кодер $C_2$. Оба кодера имеют сдвиговые регистры, содержащие по К–1 ячеек. Обратная связь обеспечивается тем, что на входы регистров с помощью сумматоров $\Sigma_1$ в каждом такте подается сумма входного символа и символов из всех ячеек регистра. На сумматоры $\Sigma_2$, так же, как и в нерекурсивном варианте кодера, подаются символы только с некоторых ячеек регистра, определяемых видом используемого порождающего полинома. Коммутатор поочередно подключает на выход символы последовательности **m** и вырабатываемых сумматорами $\Sigma_2$ последовательностей $\mathbf{v}_1$ и $\mathbf{v}_2$.

Блок чередования производит перемежение символов, изменяя порядок их следования. Последовательности символов с выходов обоих кодеров не совпадают, при этом эффективность кодирования зависит от того, как часто в обеих последовательностях будут встречаться одинаковые фрагменты. Это во многом зависит от правила перемежения. В качестве компонентных кодеров могут также использоваться линейные блоковые кодеры.

Формируемый турбокод во многом похож на выше рассмотренное произведение кодов. Отличие заключается в том, что если раньше заполнялся весь прямоугольный массив размера $n_1 \times n_2$, то теперь из него удалены $(n_1 - k_1) \times (n_2 - k_2)$ символов, т.е. все проверочные символы одного кода, полученные из проверочных символов от другого кода.

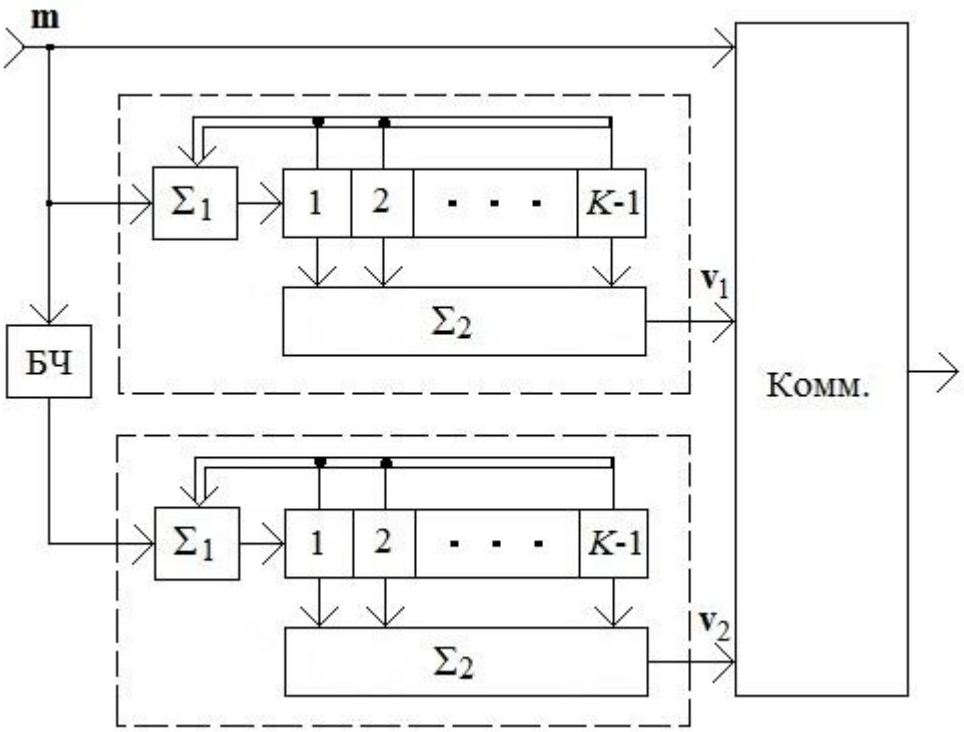



Рисунок 8.1.

## 9. Итеративно декодируемые коды и многопороговое декодирование

*Итеративно декодируемые коды*. К таким кодам обычно относят такие коды, в которых многократно используется «мягкое» декодирование, и достигаются высокие показатели помехоустойчивости при относительно небольших объемах вычислений. Кроме турбокодов и близких к ним, в настоящее время получили распространение коды с низкой плотностью проверки на четность (LDPC – low density parity control) и коды для многопорогового декодирования.

LDPC-коды – это линейные коды, у которых малое число ненулевых элементов в порождающей матрице. В принципе его можно рассматривать, как турбокод, составными кодами является совокупность простейших кодов. Во многих работах указывается, что в определенных условиях по эффективности они могут превосходить турбокоды. Регулярный LDPC-код является линейным кодом, у которого количество единичных элементов в каждой строке и столбце проверочной матрицы, связанной с порождающей матрицей, много меньше количества символов в кодовом слове (Матрица сильно разрежена, ориентировочно содержание единиц может быть порядка 1% и ниже). Кроме этого почти всегда

накладывается ограничение, состоящее в том, чтобы в матрице не было двух строк или столбцов, одновременно содержащих единицы более чем в одной позиции.

В нерегулярных LDPC-кодах распределение единичных элементов в строках и столбцах матрицы выбирается в соответствии с некоторым неравномерным распределением вероятности. Они могут обеспечить несколько бóльшую помехоустойчивость, чем регулярные коды.

*Многопороговые методы кодирования и декодирования*. Многопороговое декодирование также основано на итеративных процедурах и может быть применено как при использовании достаточно простых сверточных и блоковых кодеров, так и для сложных каскадных кодеров. Эффективность кодов сравнима с эффективностью LDPC-кодов, но практическая реализация при близких характеристиках, как правило, проще.

Количество информационных символов в блоковом коде для кодовой скорости $R=1/2$ равно как минимум $2q+1$, где $q$ – наибольшая степень порождающего полинома. Для кодовых скоростей $R=1/n$ для уменьшения риска размножения ошибок иногда от кодов вида $(n, k)$ переходят к кодам вида $(qn, qk)$, где $q$ – небольшое целое число. Общая кодовая скорость $R=qk/qn$ при этом сохраняется.

В кодере структура кодового слова образуется следующим образом. Группа символов входного информационного потока первоначально разделяется на $q$ групп и из них формируется $q$ проверочных групп символов, далее все группы передаются поочередно, складывая общее кодовое слово. Для формирования каждой проверочной группы одновременно используются все информационные группы. Пример структуры подобного кодера для $q=2$ и кодовой скорости $R=2/4$ приведен на рисунке 9.1.

Информационная группа символов $\mathbf{m}$ разбивается на две группы $\mathbf{m}_1$ и $\mathbf{m}_2$ одинаковой длины $K$ и синхронно поступает на входы 1 двух коммутаторов (Комм.). Первоначально в течение $K$ тактов оба коммутатора с выходом соединяют свои первые входы. При этом $K$ информационных символов подаются на выходы кодера и одновременно заполняют $K$ ячеек сдвиговых регистров. Далее оба коммутатора на выходы подключают свои вторые входы 2, после этого вычисляются $K$ проверочных символов двух проверочных последовательностей $\mathbf{v}_1$ и $\mathbf{v}_2$ и подаются на выходы кодера. Ко входам каждого из сумматоров по модулю 2 (Σ) подключены некоторые ячейки как первого, так и второго регистров. Номера этих ячеек определяются используемыми порождающими полиномами. Таким образом, сформированное кодовое слово образуется группами символов $\mathbf{m}_1$, $\mathbf{m}_2$, $\mathbf{v}_1$, $\mathbf{v}_2$.

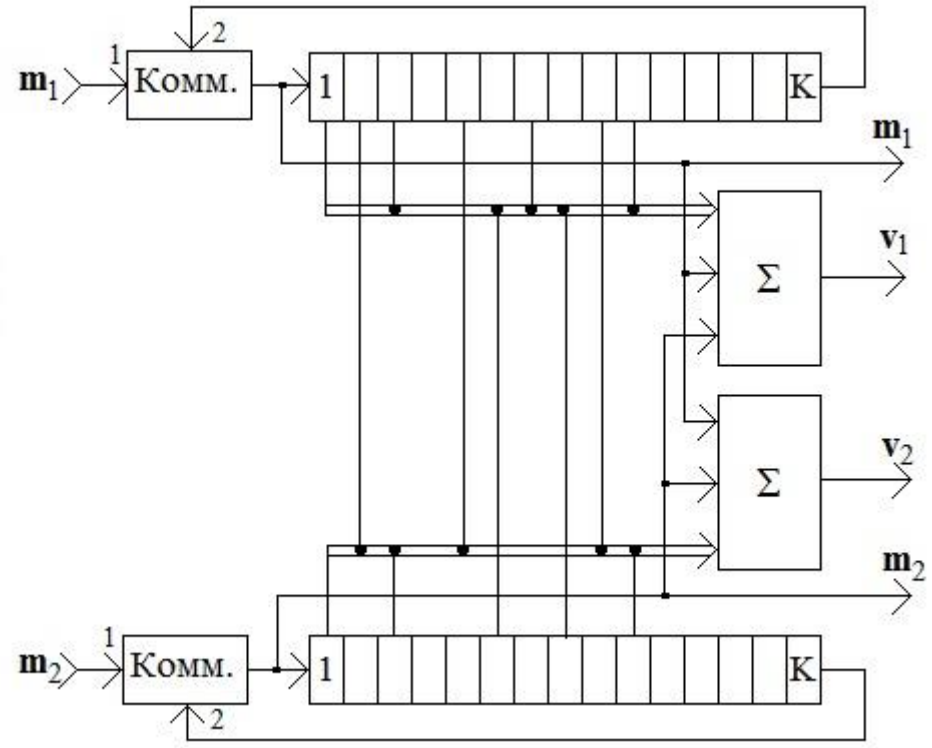



Рисунок 9.1.

## 10. Сигнально-кодовые конструкции. Перспективы развития методов кодирования

Сигнально-кодовые конструкции представляют собой совместное взаимоувязанное использование модуляции и кодирования (решетчатое кодирование – trellis coded modulation, TCM) и предназначаются для повышения эффективности использования ограниченного частотного диапазона. В случае использования цифровой модуляции сигнальное множество представляет собой набор сигналов с определенными параметрами (как правило, амплитудно-фазовыми). Каждому сигналу из набора приписывается определенное сочетание нескольких бит кодового слова. Основной результат достигается за счет следующих факторов. Переход от двоичной модуляции к использованию набора сигналов позволяет повысить скорость передачи в той же полосе частот. Однако увеличение набора сигналов снижает помехоустойчивость при ограничении предельной мощности передатчика. Однако часть ресурса увеличения скорости передачи можно потратить на введение кодирования, которое компенсирует снижение помехоустойчивости.

Этот же фактор можно использовать и при увеличении количества уже имеющихся символов в наборе. Дополнительные символы дают использовать тот или иной метод кодирования, что в свою очередь повышает помехоустойчивость передачи. При правильном

выборе параметров модуляции и кодирования этот фактор преобладает, что дополнительно повышает помехоустойчивость.

Кроме этого, большое значение имеет правильный выбор соотнесения сигналов из используемого набора и вариантов последовательности битов кодового слова. В частности, если набор сигналов представляет собой «созвездие» на фазовой плоскости, то близко расположенным сигнальным точкам присваиваются такие сочетания символов, которые из-за применения кодирования следовать одно за другим не могут. Этот фактор различного уровня вероятности ошибок дополнительно повышает помехоустойчивость.

Формирование передаваемых сигналов в подобных системах передачи информации фактически разбивается на два этапа. На первом производится собственно кодирование сигналов в их логическом представлении с учетом последующего соответствия используемому набору сигналов. А на втором этапе фрагменты кодированной последовательности символов приписываются передаваемым сигналам в их аналоговом представлении. При этом первый этап использует известные схемы кодеров.

Таким образом, современное состояние методов кодирования характеризуется большим разнообразием применяемых кодов, однако подавляющее большинство использует базовые принципы, сводящиеся к методам сверточного и блокового кодирования.

Дальнейшее развития методов кодирования связано с применением методов мало уступающих в помехоустойчивости применяемым методам, но имеющие более простую конструктивную реализацию.

## 11. Декодирование сверточных кодов

При декодировании сверточных кодов могут быть использованы различные алгоритмы. Для пояснения методов декодирования первоначально рассмотрим простейший алгоритм сверточного кодирования.

В качестве примера рассмотрим кодирование входного сообщения «110» с помощью сверточного кодера со скоростью 1/2 и *K*=3, изображенного на рисунке 11.1.

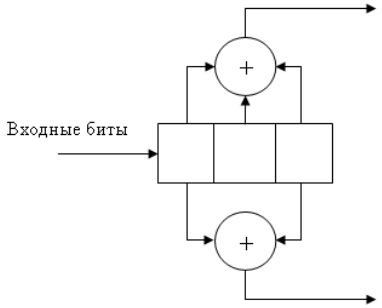



Рисунок 11.1. Сверточный кодер со степенью кодирования 1/2 и K=3

На рисунке 11.2 показан процесс кодирования данного сообщения.

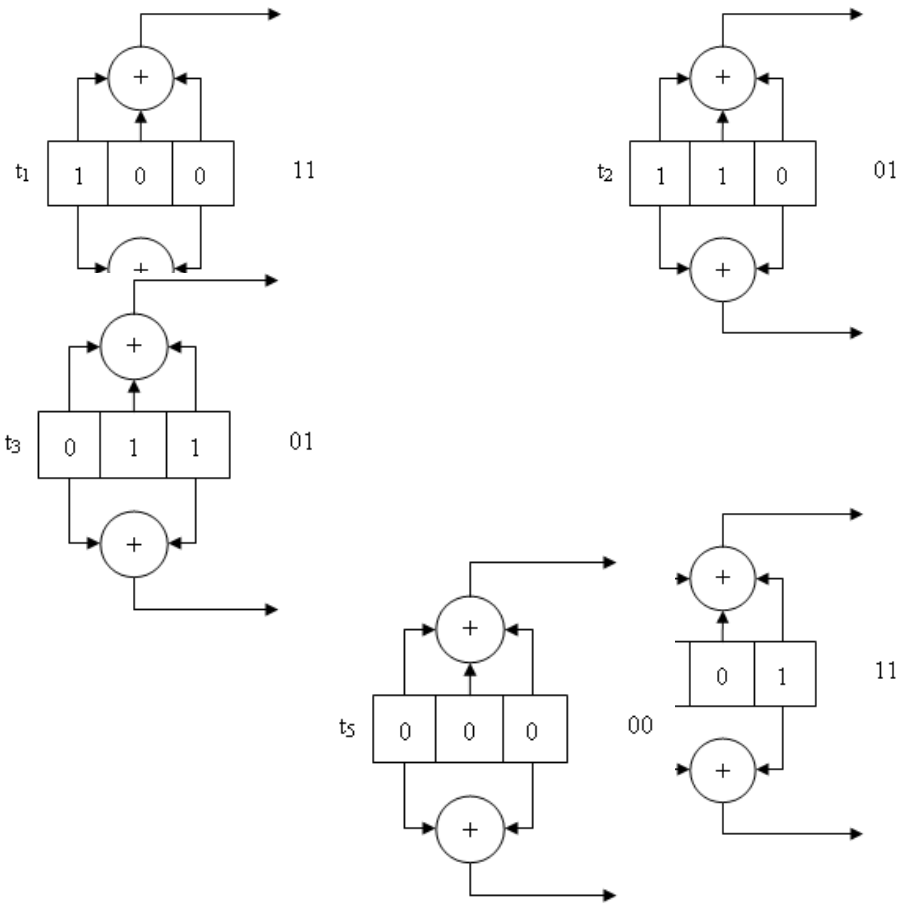



Рисунок 11.2. Кодирование сообщения «110»

Все символы по очереди поступают на вход регистра сдвига. В момент времени $t_1$ поступает символ «1». В результате операции суммирования по модулю 2 получается выходное сообщение «11». Аналогично происходит и в моменты времени $t_2$ и $t_3$. Далее вводится два нулевых входных бита для очистки регистра сдвига. В результате на выходе получается кодированная последовательность «11010111».

Любой сверточный код можно представить с помощью т.н. генератора кода или вектора связи. Генератор кода показывает наличие или отсутствие связи регистра сдвига с сумматором по модулю 2. Если на *i* позиции вектора присутствует «1», то соответствующий разряд в регистре сдвига связан с сумматором по модулю 2, а «0» указывает на отсутствие такой связи. Например, для кода, изображенного на рисунке 1.2.1.3, генератор кода для верхних и для нижних связей будет выглядеть следующим образом:

$g_1$=111;

$g_2$=101.

На практике кодовые генераторы записываются в восьмеричной системе. Рассмотренный выше код будет записываться как (7,5).

Для того, чтобы описать сверточный код, необходимо определить кодирующую функцию, то есть функцию, по которой можно по данной входной последовательности символов определить выходную последовательность. Существует несколько способов задания такой функции. Это графическая связь, полиномы связи, диаграмма состояний, древовидная и решетчатая диаграммы. Последняя является наиболее удобной и наглядной по отношению ко всем остальным. Рассмотрим принцип построения и структуру решетчатой диаграммы.

На рисунке 11.3 представлена решетчатая диаграмма, соответствующая коду (7,5).

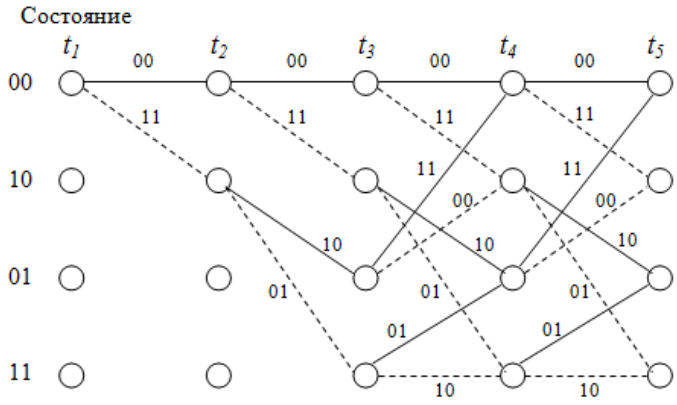



Рисунок 11.3. Решетчатая диаграмма кодера (7,5)

Решетчатая диаграмма показывает все возможные переходы кодера из предыдущего состояния в последующее. Решетка состоит из $2^{K}$-1 узлов, где $K$ – длина кодового ограничения. Каждый узел характеризует состояние кодера, то есть состояние регистра сдвига. Из каждого текущего состояния кодера можно перейти в одно из двух последующих состояний. При этом одна ветвь соответствует входному нулевому биту, а другая – входной единице. Цифры над переходом обозначают кодовые слова на выходе кодера. Сплошная линия обозначает входной «0», а пунктирная – входную «1». После достижения глубины решетки, равной $K$, она имеет периодическую структуру.

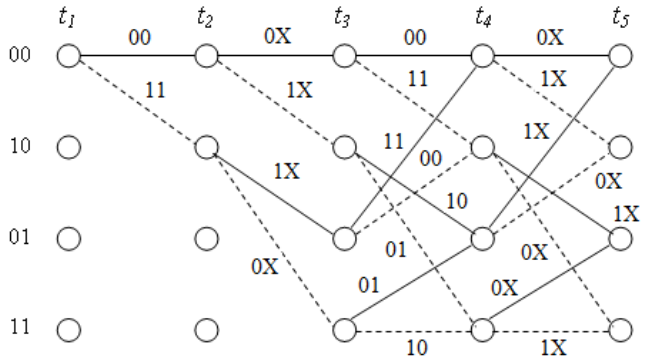
Если на выходе декодера с некоторой периодичностью проводить выкалывание, или перфорацию кодовых символов, то есть не передавать их по каналу связи, то такой код будет называться перфорированным. Как известно, наиболее простыми в реализации являются коды со скоростью $R$=1/2. Но часто бывает необходимо повысить скорость кода без изменения его структуры. Этого можно достигнуть путем перфорации символов на выходе кодера. Так как при этом структура решетки исходного низкоскоростного кода не изменяется, то с помощью одного и того же исходного кода можно получить другие высокоскоростные коды. При этом скорость кода увеличится и будет $R$=2/3. Для обозначения

правила удаления выходных символов используется матрица перфорации. Для вышеобозначенного примера матрица перфорации будет выглядеть следующим образом:

$$P = \begin{pmatrix} 5 & 5 \\ 7 & X \end{pmatrix},$$

где знак $X$ указывает местоположение вычеркнутого символа. В таблице для примера представлены различные матрицы перфорации на базе стандартного сверточного кода со скоростью $R$=1/2 и $K$=7 (код NASA).

Рассмотрим решетчатую диаграмму перфорированного сверточного кода $\begin{pmatrix} 5 & 5 \\ 7 & X \end{pmatrix}$:




Она состоит из двух повторяющихся шагов. На первом шаге присутствуют ветви, соответствующие первому столбцу матрицы перфорации. На втором шаге вычеркнутый символ не передается по каналу связи. Таким образом, декодирование сверточных кодов, полученных путем перфорации первоначального кода со скоростью 1/2 возможно декодером, который реализован согласно этой скорости.

Таблица

Матрицы перфорации для кода NASA

| Скорость кода, $R$ | Матрица перфорации |
|---|---|
| 1/2 | $\begin{pmatrix} 133 & 171 \end{pmatrix}$ |
| 2/3 | $\begin{pmatrix} 133 & 133 \\ 171 & X \end{pmatrix}$ |
| 3/4 | $\begin{pmatrix} 133 & 133 & X \\ 171 & X & 171 \end{pmatrix}$ |
| 4/5 | $\begin{pmatrix} 133 & 133 & 133 & 133 \\ 171 & X & X & X \end{pmatrix}$ |

| | |
|---|---|
| 5/6 | $\begin{pmatrix} 133 & 133 & X & 133 & X \\ 171 & X & 171 & X & 171 \end{pmatrix}$ |
| 6/7 | $\begin{pmatrix} 133 & 133 & 133 & X & 133 & X \\ 171 & X & X & 171 & X & 171 \end{pmatrix}$ |
| 7/8 | $\begin{pmatrix} 133 & 133 & 133 & 133 & X & 133 & X \\ 171 & X & X & X & 171 & X & 171 \end{pmatrix}$ |

Существует несколько способов декодирования сверточных кодов, среди которых наиболее часто используется алгоритм декодирования Витерби, который работает с решетчатым представлением и отбрасывает пути, заранее не соответствующие критерию максимального правдоподобия. Работа ведется только с выжившими путями. При этом данный алгоритм работает в соответствии с принципом максимального правдоподобия.

Алгоритм декодирования Витерби был разработан в 1967 г. с целью уменьшения объема вычислений по сравнению с алгоритмом последовательного декодирования. В 1969 г. было показано, что данный алгоритм основывается на оценке максимального правдоподобия.

Главное достоинство алгоритма Витерби заключается в том, что в нем не рассматриваются пути, которые согласно принципу максимального правдоподобия не могут быть оптимальными. Алгоритм включает в себя операции вычисления расстояния между принятым сигналом в момент времени $t_1$, и всеми путями решетки, которые входят в каждое состояние в момент времени $t_i$. Если в одно состояние входят два пути, то выбирается выживающий путь с наименьшей метрикой. В результате работы декодер постепенно проходит решетку и исключает наименее вероятные пути.

Рассмотрим работу алгоритма Витерби на примере. В качестве меры расстояния используем метрику Хэмминга. Воспользуемся кодером, изображенным на рисунке 11.1 и соответствующей ему решетчатой диаграммой, изображенной на рисунке 11.3.

Предположим, что мы имеем входную информационную последовательность m=10010. После кодирования ее сверточным кодером получаем последовательность U=1110111110, которая передается по каналу связи. В результате воздействия шума принятая последовательность будет иметь вид Z=1110011110, то есть имеет место искажение одного символа, а именно пятого бита.

На рисунке 11.4 представлена решетчатая диаграмма, в которой над каждой ветвью обозначено расстояние Хэмминга между принятым кодовым символом и кодовым словом, соответствующем данной ветви.

Рассмотрим решетку в момент времени $t_1$. Переход между состояниями 00→00 приводит к появлению на выходе кодового слова 00, но получено 11, следовательно, Хэммингово расстояние равно 2. Переход между состояниями 00→10 приводит к появлению на выходе кодового слова 11, что полностью совпадает с полученной последовательностью, и, следовательно, Хэммингово расстояние равно 0. Таким образом помечается вся решетка в последующие моменты времени.

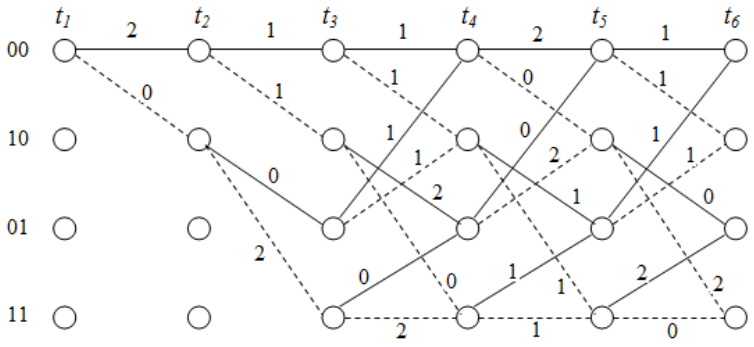
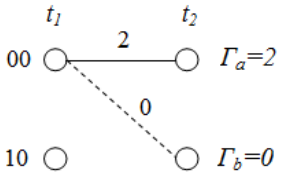


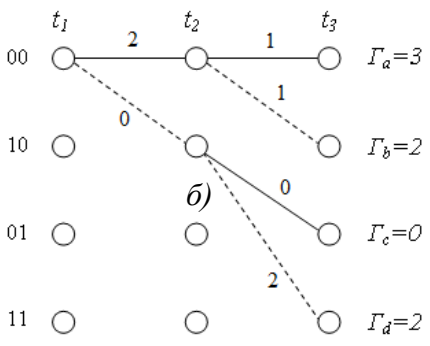

Рисунок 11.4. Решетчатая диаграмма декодера (7,5)

Теперь рассмотрим детально работу алгоритма декодирования Витерби на примере данной последовательности и кодера. Основной смысл алгоритма Витерби заключается в том, что если два пути в решетке сходятся в одной точке, то один из них при поиске оптимального пути исключается, то есть исключается путь, имеющий большую суммарную метрику пути. В случае совпадения суммарных метрик путей, путь выбирается произвольно.

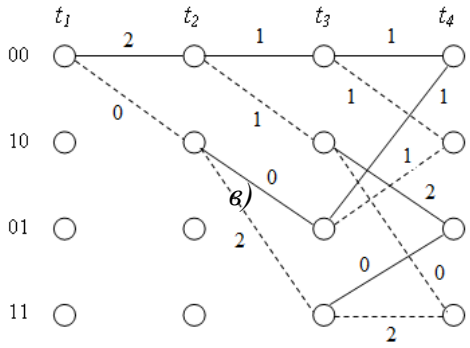
В нашем случае в момент времени $t_1$ приняты кодовые символы 11. Переходу между состояниями 00→00 соответствует метрика ветви 2, а между состояниями 00→10 метрика ветви 0. (рисунок 11.5.)



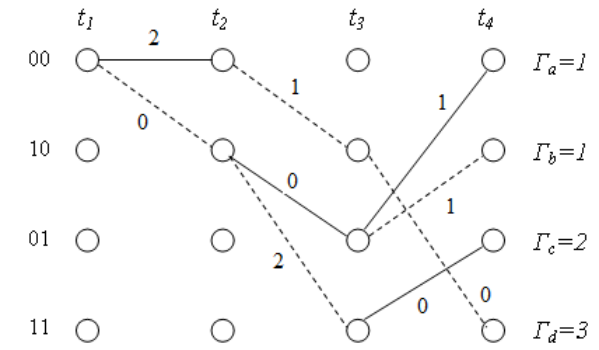

*а)*



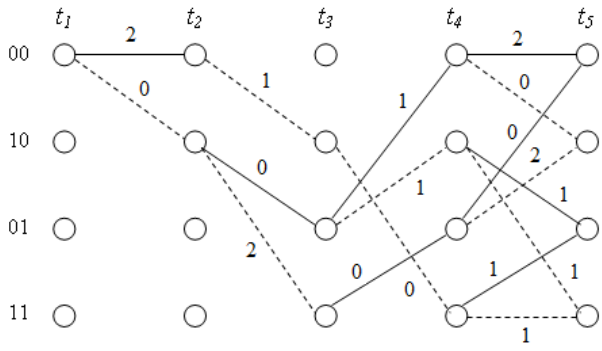

*б)*



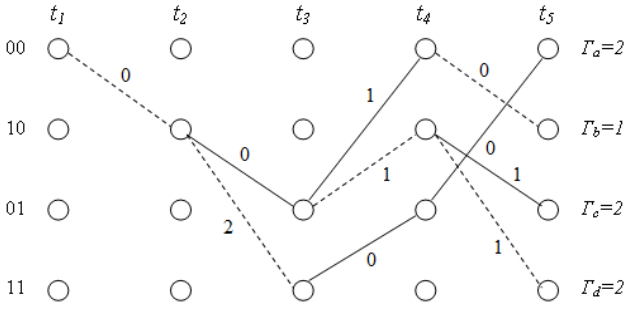

*в)*




*г)*




*д)*




*е)*

Рисунок 11.5. Пример работы алгоритма Витерби

В следующий момент времени $t_2$ из каждого предыдущего состояния выходят еще 2 ветви (рис.11.5, б). Через $\Gamma_a$, $\Gamma_b$, $\Gamma_c$ и $\Gamma_d$ обозначены суммарные метрики путей. В момент времени $t_3$ опять происходит разветвление путей (рис 11.5, в) и в момент времени $t_4$ имеется по 2 пути, входящие в каждое состояние. Путь, имеющий наибольшую суммарную метрику, может быть исключен. В результате на рисунке 11.5, г показаны выжившие пути. Та же

самая процедура происходит в момент времени $t_5$ (рисунок 11.5, д). На рис. 11.5, е показаны выжившие пути в данный момент времени. Здесь между моментами времени $t_1$ и $t_2$ остался только один выживший путь, который называется полной ветвью. При этом декодер, исходя из свойств решетки, решает, что сделан переход 00→10, которому соответствует единичный входной бит. Далее на каждом следующем шаге происходит исключение одного из путей, ведущих в каждое состояние. Таким образом, декодер будет постепенно проходить вглубь решетки, и устранять все пути кроме одного. Следует заметить, что первый бит декодируется только тогда, когда декодер углубится внутрь решетки на некоторое расстояние. Обычно оно равно 4-5 длинам кодового ограничения.

Из анализа решетчатой диаграммы сверточного кода можно сделать вывод, что существует $2^{K-2}$ непересекающихся ячейки, каждая из которых включает $2^{K-1}$ возможных перехода. Из этих ячеек и логических элементов, которые корректируют метрики состояний, и состоит декодер. Каждая ячейка реализуется с помощью логической операции, которая называется сложение, сравнение и выбор. Поскольку практическая реализация этой операции не составляет большой сложности, то очевидным преимуществом алгоритма Витерби является возможность вносить в него некоторые внутренние корректировки применительно к различным внешним условиям.

До появления алгоритма Витерби использовался алгоритм последовательного декодирования. Он был предложен Возенкрафтом и доработан Фано. Главным достоинством этого метода является способность декодировать сверточные коды с большой длиной кодового ограничения ( $K \geq 9$ ).

Сущность данного алгоритма заключается в том, что происходит обновление метрики только наиболее вероятного пути. Данная метрика рассчитывается как разность между принятым сигналом и гипотезой о переданной последовательности кодовых символов. Если достоверность пути будет ниже некоторого текущего порога, то декодер последовательно перебирает все другие пути, пока не будет найден наиболее правдоподобный путь. Таким образом, данный алгоритм можно сравнить с методом проб и ошибок для поиска правильного пути по решетке.

Главным недостатком алгоритма последовательного декодирования является зависимость числа перебираемых метрик состояний от соотношения «сигнал/шум» в канале передачи. Из-за наличия такой зависимости требуется большой объем памяти для хранения поступивших последовательностей. Иногда возникают ситуации, когда происходит переполнение буфера памяти. Так как алгоритм последовательного декодирования не является алгоритмом максимального правдоподобия, то при прочих равных условиях он уступает алгоритму Витерби.

Алгоритм декодирования с обратной связью был предложен Хеллером и основан на алгоритме порогового декодирования. Он предназначен для принятия жестких решений в двоичном симметричном канале.

Решение об информационном символе на $j$-м шаге делается исходя из метрик, вычисленных от $j$ до $j+m$ шага, где $m$ – целое положительное число. Вводится параметр – длина упреждения $L=m+1$. Она определяет количество принятых кодовых символов, которое задается для декодирования информационного бита. Решение в пользу «0» или «1» принимается исходя из минимального расстояния Хэмминга для пути, который начинается на $j$-м шаге и кончается на $j+m$ шаге, и количества «0» или «1» в ветвях, исходящих шага $j$. Часть дерева, которая не связана с решением об информационном символе, отбрасывается, а оставшаяся часть расширяется, и рассматриваются пути от шага $j+1$ до $j+1+m$. Данная процедура повторяется на каждом шаге.

Алгоритм с обратной связью имеет меньшую задержку (не более двух длин кодового ограничения) по сравнению с алгоритмом Витерби, но он принимает только жесткие решения.

Стек – алгоритм предложен Елинеком в 1969 г. Стек – алгоритм работает всего с несколькими путями. В верхней части стека располагается путь, имеющий наибольшую метрику. Далее на каждом шаге только головной путь проверяется по разветвлению. В результате образуется $2^K$ продолжений путей, которые затем вместе с другими путями упорядочиваются согласно с величинами их метрик. Затем все пути, значения метрик которых располагаются ниже некоторой величины от метрики главного пути, отбрасываются, и процесс продолжения путей с наибольшими метриками вновь повторяется.

По сравнению с алгоритмом Витерби, стек – алгоритм требует меньшего числа сравнений метрик, но требуется большая вычислительная способность для его реализации.

## 12. Декодирование блоковых кодов

Рассмотрим декодирование блоковых циклических кодов, как самого распространенного вида используемых блоковых кодов. При «жестком» декодировании на уровне демодулятора все принимаемые символы считаются взаимно независимыми, поэтому каждый из них независимо декодируется по принципу максимального правдоподобия. А уже после этого декодер, использую определенные связи между всей совокупностью символов блока, которая была введена при кодировании за счет использования избыточности, исправляет возможные ошибки в информационной части бока.

В случае «мягкого» декодирования реализация принципа максимального правдоподобия тоже потребует для каждого информационного символа выбора на основе принимаемой реализации его наиболее вероятного значения. Однако при этом проверочные символы несут дополнительную информацию о значении информационных символов и могут быть использованы для *коррекции* тех исходных вероятностей информационных символов, которые ранее определил демодулятор. В результате эти вероятности изменятся, и наиболее вероятными могут оказаться другие значения символов.

Такими образом, «мягкое» декодирование можно трактовать, как коррекцию исходных величин вероятности принимаемых символов, определенных демодулятором, с последующим выбором их наиболее вероятных значений на основе откорректированных величин вероятностей. Подобный подход позволяет значительно уменьшить требуемый объем вычислений.

Рассмотрим реализацию предлагаемого алгоритма.

Как известно, при «жестком» декодировании для обнаружения и исправления ошибок с использованием циклического кода наиболее общий алгоритм состоит из следующей последовательности операций:

- Принятый блок – кодовая комбинация делится на образующий многочлен g(x). Если остаток от деления $R(x)\neq 0$, то определяется вес остатка w (количество единиц в остатке). Если вес остатка равен или меньше числа исправляемых ошибок t ($w\leq t$), то принятую комбинацию можно сложить по модулю 2 с остатком и получить исправленную комбинацию. Однако можно этого и не делать, т.к. при этом будут откорректированы только проверочные символы, которые после детектирования ценности обычно не представляют. Информационную же часть блока при этом можно считать свободной от ошибок.

- Если $w>t$ , то производится циклический сдвиг на один символ влево. Полученная после такого сдвига комбинация снова делится на образующий многочлен g(x). Если при этом вес остатка получился $w\leq t$, то эту циклически сдвинутую комбинацию складывают по mod2 с полученным остатком. Затем после этого сложения его результат циклически сдвигают обратно (вправо) на один символ, т.е. восстанавливают ее исходный вид. При этом также восстанавливается и исправленная информационная часть.

- Если после предыдущего циклического сдвига на один символ после деления на образующий полином g(x) вновь сохраняется неравенство $w>t$, производят следующие посимвольные циклические сдвиги влево. При этом после каждого сдвига осуществляется деление сдвинутой комбинации на g(x) и анализируется вес остатка. Когда после какого-либо сдвига начинает выполняться условие $w\leq t$, то сдвинутую комбинацию складывают с

полученным остатком, после чего производят обратных циклических сдвигов вправо столько, сколько их было сделано влево.

В случае использования недвоичных блоковых кодов декодирование осуществляется с использованием полей Галуа, т.е. алгебры конечных элементов.