

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ В.Ф. УТКИНА

Кафедра радиоуправления и связи

МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ

по дисциплине (модулю)

«Методы первичного кодирования в телекоммуникациях»

Направление подготовки

11.04.02 «Инфокоммуникационные технологии и системы связи»

Направленность (профиль) подготовки

Интеллектуальные системы и сети телекоммуникаций

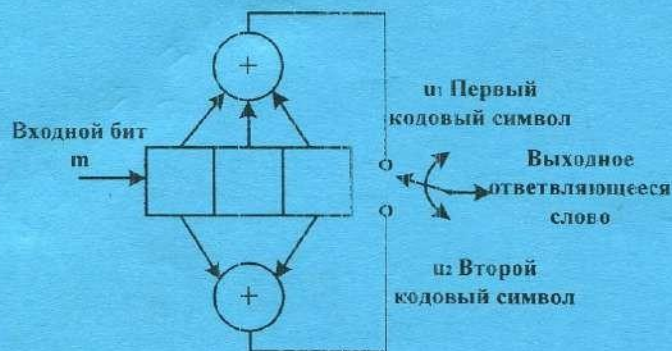
Квалификация выпускника – Магистр

Форма обучения – заочная

Рязань 2025

КОДИРОВАНИЕ И ДЕКОДИРОВАНИЕ СВЕРТОЧНЫХ КОДОВ

Методические указания
к лабораторным работам № 3 и 4



Рязань 2023

УДК 621.396.43

Кодирование и декодирование сверточных кодов: методические указания к лабораторным работам № 3 и 4/ Рязан. гос. радиотехн.ун-т; сост.: М.В. Кулакова, А.В. Егоров. Рязань, 2023. 24 с.

Изложены методы кодирования и декодирования сверточных кодов.

Предназначены для студентов 4-го курса дневной формы обучения специальности 11.05.01 «Радиоэлектронные системы и комплексы».

Табл. 4. Ил. 15. Библиогр.: 2 назв.

Сверточный код, регистр сдвига, сумматор по модулю 2, коммутатор, скорость кода, память кода, метрика, декодирование Витерби

Печатается по решению редакционно-издательского совета Рязанского государственного радиотехнического университета.

Рецензент: кафедра радиоуправления и связи Рязанского государственного радиотехнического университета (зав. кафедрой канд. техн. наук, доц. В.Т. Дмитриев)

Кодирование и декодирование сверточных кодов

Составители: Кулакова Марина Васильевна
Егоров Алексей Владимирович

Редактор Р.К. Мангутова

Корректор С.В. Макушина

Подписано в печать 27.04.23. Формат бумаги 60x84 1/16.

Бумага писчая. Печать трафаретная. Усл. печ. л. 1,5.

Тираж 50 экз. Заказ 4242.

Рязанский государственный радиотехнический университет.

390005, Рязань, ул. Гагарина, 59/1.

Редакционно-издательский центр РГРТУ.

1. Общие сведения

Сверточные коды (СК) относятся к непрерывным кодам. Здесь нет деления на кодовые комбинации, как в блочных. Выходные элементы в данном случае зависят от ряда предшествующих информационных элементов. Сверточные коды являются помехоустойчивыми, т.е. позволяют обнаруживать и исправлять некоторые ошибки. Это возможно за счет избыточности.

Основными элементами сверточного кода являются: регистр сдвига, сумматор по модулю 2, коммутатор.

Регистр сдвига — это динамическое запоминающее устройство, хранящее двоичные символы 0 и 1. Память кода определяет число триггерных ячеек m в регистре сдвига. Когда на вход регистра сдвига поступает новый информационный символ, то символ, хранящийся в крайнем правом разряде, выводится из регистра и сбрасывается. Остальные символы перемещаются на один разряд вправо, и, таким образом, освобождается крайний левый разряд, куда будет поступать новый информационный символ.

Сумматор по модулю 2 осуществляет сложение поступающих на него символов 1 и 0. Правило сложения по модулю 2 таково: сумма двоичных символов равна 0, если число единиц среди поступающих на входы символов четно, и равно 1, если это число нечетно.

Коммутатор последовательно считывает поступающие на его входы символы и устанавливает на выходе очередность кодовых символов в канал связи.

Сверточный код описывается тремя целыми числами (n, k, K) . В данном случае k — число информационных символов, поступающих за один такт на вход кодера, n — число символов на выходе кодера, соответствующих k поступившим на вход символам в течение такта, K — длина кодового ограничения (*память кода*), она определяется числом разрядов (ячеек памяти) в кодирующем регистре сдвига. Кодовое ограничение определяет мощность и сложность кода. Мы будем рассматривать только наиболее часто используемые двоичные сверточные кодеры, для которых $k = 1$, т.е. те кодирующие устройства, в которых биты сообщения сдвигаются по одному биту за раз.

Выходные элементы СК зависят не только от текущего входного элемента, но и от $(K-1)$ предыдущих, т.е. СК имеет память.

Под скоростью кода в теории кодирования понимается отношение $R = k/n$. Более точное наименование параметра R — относительная скорость кода, поскольку за единицу времени кодер принимает на

вход k информационных разрядов и трансформирует их в n разрядов избыточного кода.

При изучении сверточных кодеров в качестве модели будем использовать сверточный кодер, показанный на рис. 1. На этом рисунке изображен сверточный кодер (2,1) с длиной кодового ограничения $K = 3$. В нем имеется $n = 2$ сумматора по модулю 2; следовательно, скорость кодирования кода k/n равна $1/2$. При каждом такте бит помещается в крайний левый разряд, а биты регистра сдвигаются на одну позицию вправо. Затем коммутатор на выходе дискретизирует выходы всех сумматоров по модулю 2 (т.е. сначала верхнего сумматора, затем нижнего), в результате чего формируются пары кодовых символов, образующих слово, связанное с только что поступившим битом. Это выполняется для каждого входного бита. Выбор связи между сумматорами и разрядами регистра влияет на характеристики кода. Всякое изменение в выборе связей приводит в результате к различным кодам.

В отличие от блочных кодов, имеющих фиксированную длину слова n , в сверточных кодах нет определенного размера блока. Однако с помощью *периодического отбрасывания* сверточным кодам часто принудительно придают блочную структуру. Это требует некоторого количества нулевых разрядов, присоединенных к концу входной последовательности данных, которые служат для очистки регистра сдвига от бит данных.

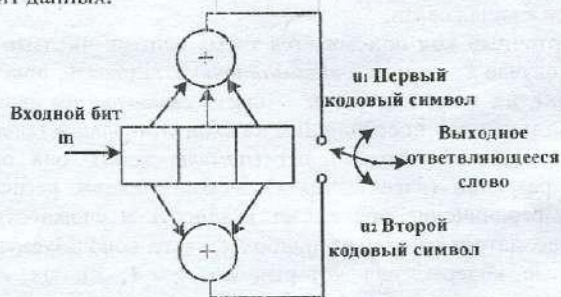


Рис. 1. Сверточный кодер (степень кодирования $1/2$, $K = 3$)

Один из способов реализации кодера заключается в определении n векторов связи, по одному на каждый из n сумматоров по модулю 2. Каждый вектор имеет размерность K и описывает связь регистра сдвига кодера с соответствующим сумматором по модулю 2. Единица на i -й позиции вектора указывает на то, что соответствующий разряд в регистре сдвига связан с сумматором по модулю 2, а ноль в

данной позиции указывает, что связи между разрядом и сумматором по модулю 2 не существует. Для кодера на рис. 1 можно записать вектор связи g_1 для верхних связей, а g_2 — для нижних.

$$g_1 = 111$$

$$g_2 = 101$$

Предположим теперь, что входная последовательность $m = 101$ закодирована с использованием сверточного кода и кодера, показанного на рис. 1. Введены три бита сообщения, по одному в момент времени t_1, t_2, t_3 , как показано на рис. 2. Затем для очистки регистра в моменты времени t_4 и t_5 введены $(K-1) = 2$ нуля, что в результате приводит к смещению конечного участка на всю длину регистра. Последовательность на выходе выглядит следующим образом: 1110001011, где крайний левый символ представляет первую передачу. В момент времени t_6 показан нулевой выход, это должно дать возможность убедиться в том, что в момент времени t_5 регистр устанавливается в исходное состояние. Таким образом, в момент времени t_6 уже можно передавать новое сообщение.

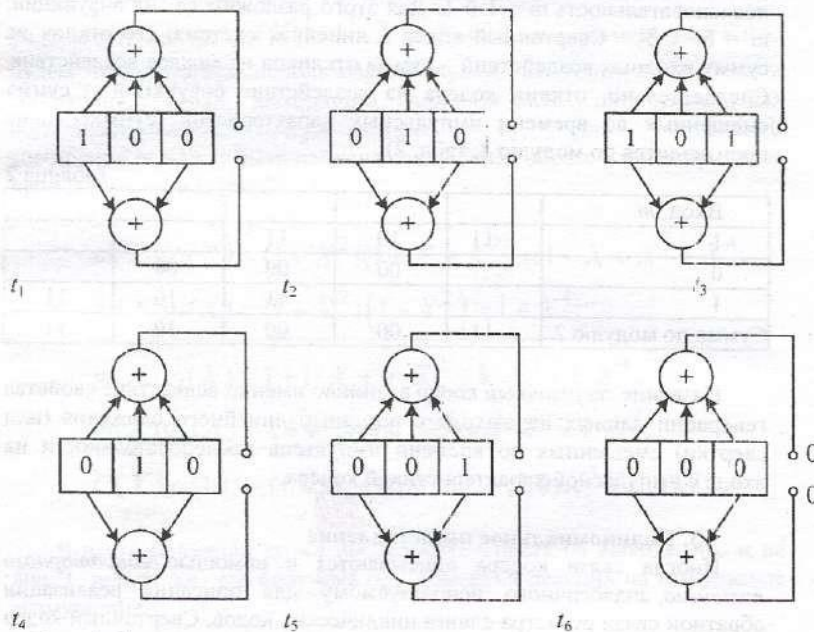


Рис. 2. Сверточное кодирование последовательности сообщения со степенью кодирования $1/2$ кодером с $K=3$

2. Реакция кодера на импульсное возмущение

Можно описать кодер через его *импульсную характеристику*, т.е. в виде отклика кодера на единичный проходящий бит (δ -функцию). Рассмотрим содержимое регистра (табл. 1) при прохождении через него двоичной единицы, т.е. входная последовательность: 1 0 0.

Таблица 1

Содержимое регистра	u_1	u_2
100	1	1
010	1	0
001	1	1

Выходная последовательность для кодера, представленного на рис. 1, имеет вид 11 10 11.

Последовательность на выходе при единице на входе называется откликом кодера на импульсное возмущение, или его импульсной характеристикой. Найдем отклик кодера на входную последовательность $m = 1\ 0\ 1$. Для этого разложим ее на δ -функции: $m = \delta_1 + \delta_3$. Сверточный кодер – линейная система; его отклик на сумму входных воздействий – сумма откликов на каждое воздействие. Следовательно, отклик кодера на воздействие δ -функций – сумма смещенных во времени импульсных характеристик (суммирование производится по модулю 2, табл. 2).

Таблица 2

Вход, m					
1	11	10	11		
0		00	00	00	
1			11	10	11
Сумма по модулю 2	11	00	00	10	11

Название *сверточный кодер* возникло именно вследствие свойства генерации данных на выходе с помощью линейного сложения (или свертки) смещенных во времени импульсов последовательности на входе с импульсной характеристикой кодера.

3. Полиномиальное представление

Иногда связи кодера описываются с помощью *генераторного полинома*, аналогичного используемому для описания реализации обратной связи регистра сдвига циклических кодов. Сверточный кодер можно представить в виде набора из n полиномиальных генераторов, по одному для каждого из n сумматоров по модулю 2.

Каждый полином описывает связь кодирующего регистра сдвига с соответствующим сумматором по модулю 2, почти так же, как и вектор связи. Коэффициенты возле каждого слагаемого полинома порядка $(K - 1)$ равны либо 1, либо 0, в зависимости от того, имеется ли связь между регистром сдвига и сумматором по модулю 2. Для кодера на рис. 1 можно записать полиномиальный генератор $g_1(X)$ для верхних связей и $g_2(X)$ — для нижних:

$$g_1(X) = 1 + X + X^2;$$

$$g_2(X) = 1 + X.$$

Здесь слагаемое самого нижнего порядка в полиноме соответствует входному разряду регистра. Выходная последовательность находится следующим образом:

$$U(X) = m(X)g_1(X) \text{ чередуется с } m(X)g_2(X).$$

Прежде всего, выразим входную последовательность $m = 101$ в виде полинома, т.е. $m(X) = 1 + X^2$. Для очистки регистра мы снова будем предполагать использование нулей, следующих за битами сообщения. Тогда выходящий полином $U(X)$, или выходящая последовательность U кодера (рис. 1) для входящего сообщения m может быть найдена следующим образом (суммирование происходит по модулю 2):

$$m(X)g_1(X) = (1 + X^2)(1 + X + X^2) = 1 + X + X^3 + X^4;$$

$$m(X)g_2(X) = (1 + X^2)(1 + X) = 1 + X^3;$$

$$m(X)g_1(X) = 1 + 1 \cdot X + 0 \cdot X^2 + 1 \cdot X^3 + 1 \cdot X^4;$$

$$m(X)g_2(X) = 1 + 0 \cdot X + 0 \cdot X^2 + 0 \cdot X^3 + 1 \cdot X^4;$$

$$U(X) = (1, 1) + (1, 0)X + (0, 0)X^2 + (1, 0)X^3 + (1, 1)X^4.$$

В итоге получили ту же последовательность на выходе, что и на рис. 1, или ту же, что получена при описании реакции на импульсное возмущение.

4. Представление состояния и диаграмма состояний

Одним из способов представления простых кодирующих устройств является *диаграмма состояния* (state diagram); такое представление кодера, изображенного на рис. 1, показано на рис. 3. Состояния, показанные в рамках диаграммы, представляют собой возможное содержимое $K-1$ крайних правых разрядов регистра, а пути между состояниями — ответвляющиеся слова на выходе, являющиеся результатом переходов между такими состояниями. Состояния регистра выбраны следующими: $a = 00$, $b = 10$, $c = 01$ и $d = 11$. Диаграмма, показанная на рис. 3, иллюстрирует все возможные смены состояний для кодера, показанного на рис. 1. Существует всего два исходящих из каждого состояния перехода, соответствующих двум возможным входным битам. Далее для каждого пути между состояниями записано ответвляющееся слово на выходе, связанное с переходами между состояниями. При изображении путей сплошной линией принято обозначать путь, связанный с единичным входным битом, а пунктирной линией — путь, связанный с нулевым входным битом. Отметим, что за один переход невозможно перейти из данного состояния в любое произвольное. Так как за единицу времени перемещается только один бит, существует только два возможных перехода между состояниями, в которые регистр может переходить за время прохождения каждого бита. Например, если состояние кодера — 00, при следующем смещении возможно возникновение только состояний 00 или 10.



Рис. 3. Диаграмма состояний кодера (скорость кодирования $1/2$, $K=3$)

Для кодера, показанного на рис. 1, найдем изменение состояний и результирующую последовательность кодовых слов U для входной последовательности $m = 1\ 1\ 0\ 1\ 1$, за которой следует $K-1 = 2$ нуля для очистки регистра. Предполагается, что в исходном состоянии регистр содержит одни нули (табл. 3).

Таблица 3

Входные биты, m_i	Содержимое регистра	Состояние в момент времени t_i	Состояние в момент времени t_{i+1}	Ответвляющееся слово в момент времени t_i	
				u_1	u_2
-	000	00	00	-	-
1	100	00	10	1	1
1	110	10	11	0	1
0	011	11	01	0	1
1	101	01	10	0	0
1	110	10	11	0	1
0	011	11	01	0	1
0	001	01	00	1	1

Последовательность на выходе $U = 11\ 01\ 01\ 00\ 01\ 01\ 11$.

5. Решетчатая диаграмма

Решетчатая диаграмма, которая использует повторяющуюся структуру, дает более удобное описание кодера. Решетчатая диаграмма для сверточного кодера, изображенного на рис. 1, показана на рис. 4.

При изображении решетчатой диаграммы мы воспользовались теми же условными обозначениями, что и для диаграммы состояния: сплошная линия обозначает выходные данные, генерируемые входным единичным битом, а пунктирная — выходные данные, генерируемые входным нулевым битом. Узлы решетки представляют состояния кодера; первый ряд узлов соответствует состоянию $a = 00$, второй и последующие — состояниям $b = 10$, $c = 01$ и $d = 11$. В каждый момент времени для представления 2^{K-1} возможных состояний кодера решетка требует 2^{K-1} узлов. В нашем примере после достижения глубины решетки, равной трем (в момент времени t_4), замечаем, что решетка имеет фиксированную периодическую структуру. В общем случае фиксированная структура реализуется после достижения глубины K . Следовательно, с этого момента в каждое состояние можно войти из любого из двух предыдущих состояний. Также из каждого состояния можно перейти в одно из двух состояний. Из двух

исходящих ветвей одна соответствует нулевому входному биту, а другая — единичному входному биту.

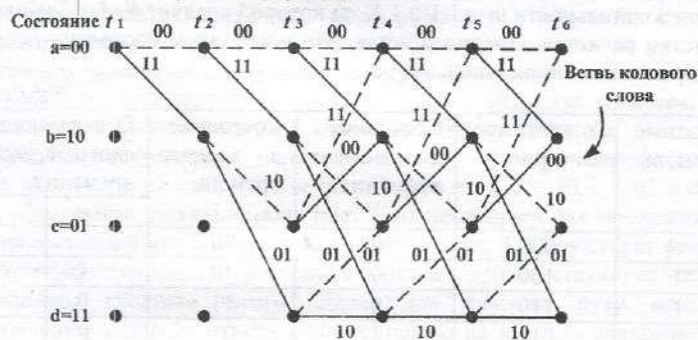


Рис. 4. Решетчатая диаграмма кодера

На рис. 4 отвечающие слова на выходе соответствуют переходам между состояниями, показанными как метки на ветвях решетки. Состояние сверточного кодера представлено содержанием крайних правых $K-1$ разрядов в регистре кодера.

На рис. 5 приведен пример работы кодера, представленного на рис. 1, при кодировании входной последовательности $m = 10110100$.

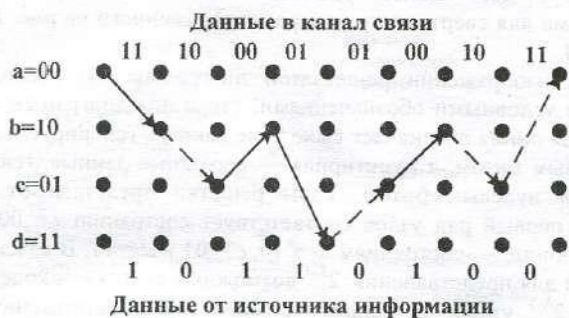


Рис. 5. Путь на решетчатой диаграмме сверточного кода

6. Алгоритм сверточного декодирования Витерби

Алгоритм декодирования Витерби был открыт и проанализирован Витерби в 1967 году. Алгоритм включает в себя

вычисление *расстояния* (или *метрику*) между сигналом, полученным в момент времени t_1 , и всеми путями решетки, входящими в каждое состояние в момент времени t_1 . Если в одно и то же состояние входят два пути, выбирается тот, который имеет лучшую метрику; такой путь называется *выживающим*. Отбор выживающих путей выполняется для каждого состояния. Таким образом, декодер углубляется в решетку, принимая решения путем исключения менее вероятных путей. Предварительный отказ от маловероятных путей упрощает процесс декодирования. Отметим, что задачу отбора оптимальных путей можно выразить как выбор кодового слова с *максимальной метрикой правдоподобия* или *минимальной метрикой расстояния*.

7. Пример сверточного декодирования Витерби

В данном случае приемлемой мерой расстояния будет расстояние Хэмминга. Расстояние Хэмминга — число позиций, в которых различаются соответствующие символы двух строк одинаковой длины. Кодер для этого примера показан на рис. 1, а решетчатая диаграмма — на рис. 4. Для представления декодера, как показано на рис. 6, можно воспользоваться подобной решеткой. Мы начинаем в момент времени t_1 в состоянии 00 (вследствие очистки кодера между сообщениями декодер находится в начальном состоянии). Поскольку в этом примере возможны только два перехода, начинающиеся в некотором состоянии, для начала не нужно показывать все ветви. Полная решетчатая структура образуется после момента времени t_3 . Принцип работы происходящего после процедуры декодирования можно понять, изучив решетку кодера на рис. 4 и решетку декодера, показанную на рис. 6.

Входная

информационная

последовательность m 1 1 0 1 1

Переданные

словесные слова U 11 01 01 00 01

Принятая

последовательность Z 11 01 01 10 01

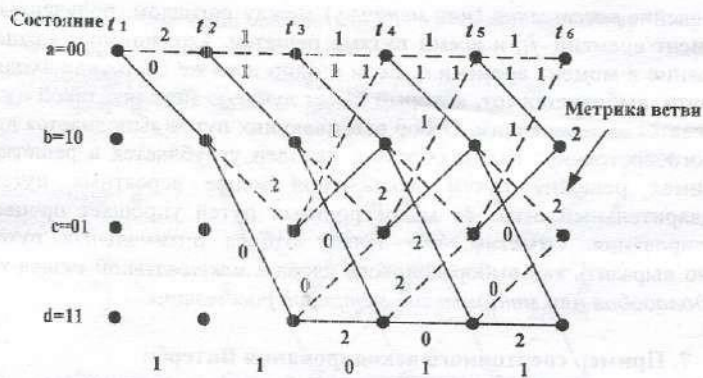


Рис. 6. Решетчатая диаграмма декодера

Для решетки декодера каждую ветвь за каждый временной интервал удобно пометить *расстоянием Хэмминга* между полученным кодовым символом и отвечающим словом, соответствующим той же ветви из решетки кодера. На рис. 6 показана последовательность сообщений m , соответствующая последовательности кодовых слов U . Как показано на рис. 4, кодер характеризуется кодовыми словами, находящимися на ветвях решетки кодера и заведомо известными как кодеру, так и декодеру. Эти отвечающие слова являются кодовыми символами, которые можно было бы ожидать на выходе кодера в результате каждого перехода между состояниями. Пометки на ветвях решетки декодера накапливаются в процессе декодирования. Другими словами, когда получен кодовый символ, каждая ветвь решетки декодера помечена метрикой подобия (расстоянием Хэмминга) между полученным кодовым символом и каждым отвечающим словом за этот временной интервал. Из полученной последовательности Z , показанной на рис. 6, можно видеть, что кодовые символы, полученные в момент времени t_1 , — это 11. Чтобы пометить ветви декодера подходящей метрикой расстояния Хэмминга в момент времени t_1 , рассмотрим решетку кодера на рис. 4. Видим, что переход между состояниями $00 \rightarrow 00$ порождает на выходе отвечающее слово 00. Однако получено 11. Следовательно, на решетке декодера помечаем переход между состояниями $00 \rightarrow 00$ расстоянием Хэмминга между ними, а именно 2. Глядя вновь на решетку кодера, видим, что переход между состояниями $00 \rightarrow 10$ порождает на выходе отвечающее слово 11, точно

соответствующее полученному в момент t_1 кодовому символу. Следовательно, переход на решетке декодера между состояниями $00 \rightarrow 10$ помечаем расстоянием Хэмминга 0. В итоге метрика входящих в решетку декодера ветвей описывает разницу (расстояние) между тем, что было получено, и тем, что "могло бы быть" получено, при наличии ответвленных слов, связанных с теми ветвями, с которых они были переданы. По сути, эти метрики описывают величину, подобную корреляциям между полученным ответвляющимся словом и каждым из кандидатов на роль отвечающего слова. Таким же образом продолжаем помечать ветви решетки декодера по мере получения символов в каждый момент времени t_i . В алгоритме декодирования эти метрики расстояния Хэмминга используются для нахождения *наиболее вероятного* (с минимальным расстоянием) пути через решетку.

Смысл декодирования Витерби заключается в следующем. Если любые два пути сливаются в одном состоянии, то при декодировании вычисляются метрики этих путей и оставляется один – выживший путь с минимальной метрикой.

На рис. 7 показаны два пути, сливающихся в момент времени t_5 в состоянии 00. Определим суммарную метрику пути по Хэммингу для данного пути в момент времени t_5 , как сумму метрик ветвей, по которым проходит путь. Верхний путь имеет метрику 4, нижний – метрику 1. Верхний путь нельзя выделить как оптимальный, поскольку нижний путь, входящий в то же состояние, имеет меньшую метрику. Именно этот путь считается выжившим.

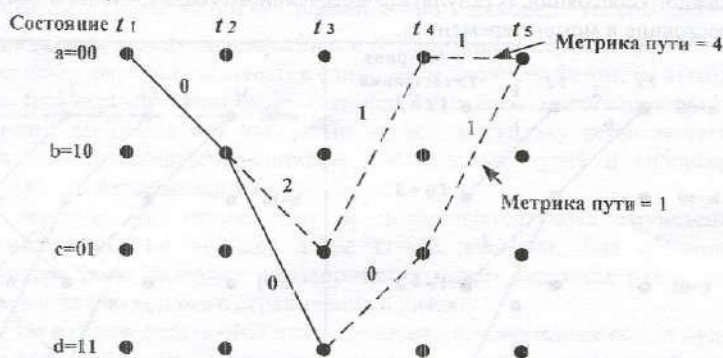


Рис. 7. Метрики пути для двух сливающихся путей

Опишем несколько первых шагов в рассматриваемом примере декодирования (рис. 8). Предположим, что последовательность входящих данных m , кодовое слово U и полученная последовательность Z аналогичны приведенным на стр. 9, а исходное состояние кодера — 00. В момент времени t_1 получены кодовые символы 11. Из состояния 00 можно перейти только в состояние 00 или 10, как показано на рис. 8.

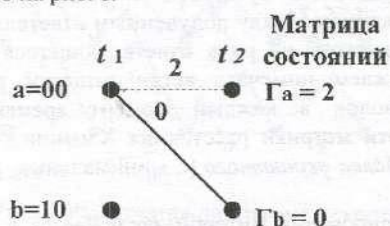


Рис. 8. Пути, выжившие на момент t_2

Переход между состояниями $00 \rightarrow 10$ имеет метрику ветви 0; переход между состояниями $00 \rightarrow 00$ — метрику ветви 2. На момент времени t_2 из каждого состояния также может выходить только две ветви, как показано на рис. 9. Суммарная метрика этих ветвей обозначена как матрица состояний $\tilde{A}_a, \tilde{A}_b, \tilde{A}_c$ и $\tilde{A}_d$, соответствующих конечным состояниям.

В момент времени t_3 на рис. 10 опять есть две ветви, выходящие из каждого состояния. В результате имеется два пути, входящих в каждое состояние в момент времени t_4 .

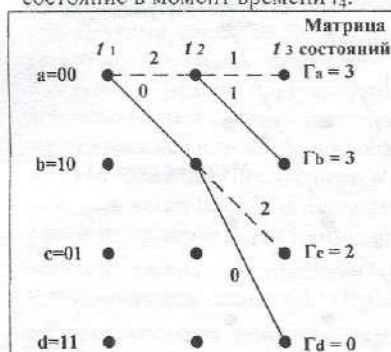


Рис. 9. Пути, выжившие на момент t_3

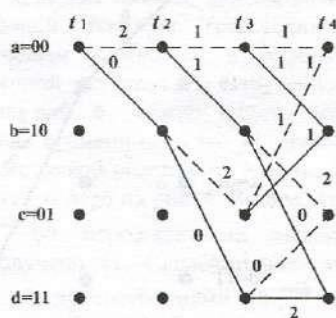


Рис. 10. Сравнение метрик на момент t_4

Один из путей, входящих в каждое состояние, может быть исключен, а точнее — это путь, имеющий большую суммарную метрику пути. Если бы метрики двух входящих путей имели одинаковое значение, то путь, который будет исключаться, выбирался бы произвольно. Выживший путь в каждом состоянии показан на рис. 11.

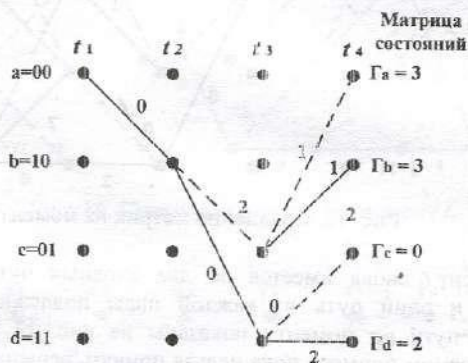
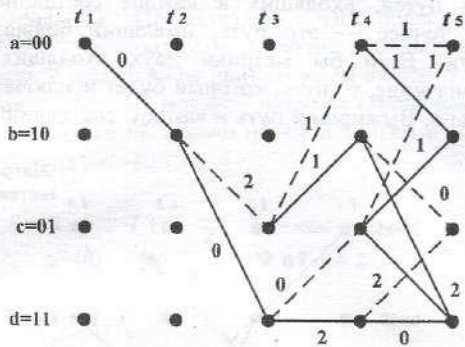


Рис. 11. Пути, выжившие на момент t_4

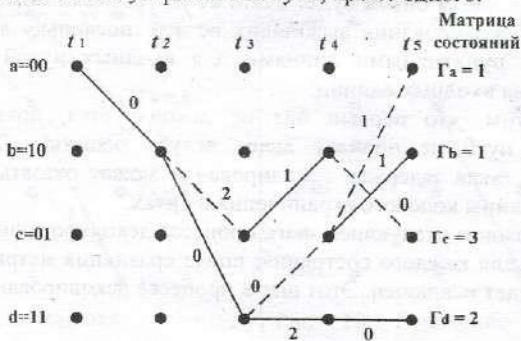
В этой точке процесса декодирования имеется только один выживший путь, который называется *полной ветвью*, между моментами времени t_1 и t_2 . Следовательно, декодер теперь может решить, что между моментами t_1 и t_2 произошел переход $00 \rightarrow 10$. Поскольку переход вызывается единичным входным битом, на выходе декодера первым битом будет единица. Здесь легко можно проследить процесс декодирования выживших ветвей, поскольку ветви решетки показаны пунктирными линиями для входных нулей и сплошной линией для входных единиц.

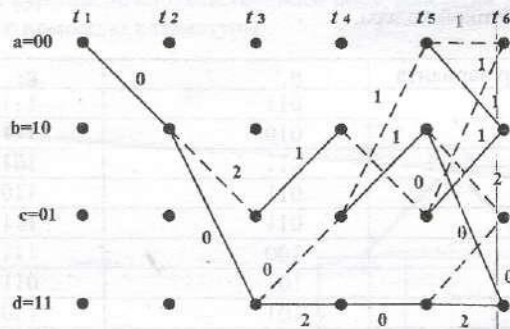
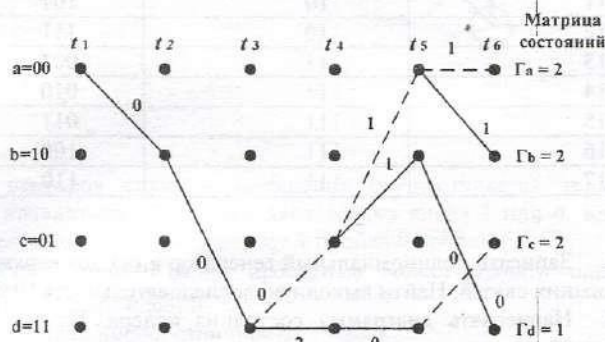
Заметим, что первый бит не декодируется, пока вычисление метрики пути не пройдет далее вглубь решетки. Для обычного декодера такая задержка декодирования может оказаться раз в пять больше длины кодового ограничения в битах.

На каждом следующем шаге процесса декодирования всегда будет два пути для каждого состояния; после сравнения метрик путей один из них будет исключен. Этот шаг в процессе декодирования показан на рис. 12.

Рис. 12. Сравнение метрик на момент t_5

В момент t_5 снова имеется по два входных пути для каждого состояния, и один путь из каждой пары подлежит исключению. Выжившие пути на момент t_5 показаны на рис. 13. Заметим, что в рассматриваемом примере пока нельзя принять решение относительно второго входного информационного бита, поскольку еще остается два пути, исходящих в момент t_2 из состояния в узле 10. В момент времени t_6 на рис. 14 снова можем видеть структуру сливающихся путей, а на рис. 15 — выжившие пути на момент t_6 . Здесь же, на рис. 15, на выходе декодера в качестве второго декодированного бита показана единица как итог единственного оставшегося пути между точками t_2 и t_3 . Аналогичным образом декодер продолжает углубляться в решетку и принимать решения, касающиеся информационных битов, устраняя все пути, кроме одного.

Рис. 13. Пути, выжившие на момент t_5

Рис. 14. Сравнение метрик на момент t_6 Рис. 15. Пути, выжившие на момент t_6

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Домашнее задание.

1. Реализовать кодер для заданных векторов связи g_1 и g_2 (значения g_1 и g_2 взять из табл. 4). Номер варианта соответствует номеру студента в списке группы.
2. Задать входную последовательность m , состоящую из 5 символов. Закодировать сообщение с помощью реализованного кодера.
3. Найти для заданной входной последовательности m данные на выходе кодера путем линейного сложения смещенных во времени

импульсов последовательности на входе с импульсной характеристикой кодера.

Таблица 4

Номер варианта	g_1	g_2
1	011	111
2	010	111
3	011	101
4	011	110
5	011	111
6	100	111
7	101	011
8	101	110
9	101	111
10	110	011
11	110	101
12	110	111
13	111	001
14	111	010
15	111	011
16	111	100
17	111	110

4. Записать полиномиальный генератор $g_1(x)$ для верхних и $g_2(x)$ для нижних связей. Найти выходную последовательность $U(x)$.
5. Нарисовать диаграмму состояния кодера. Найти изменение состояний и результирующую последовательность слов U для заданной последовательности сообщений m .
6. Нарисовать решетчатую диаграмму кодера.
7. С помощью метода декодирования Витерби получить из закодированного сообщения несколько символов переданного сообщения.
8. Внести ошибку в принятое сообщение. Доказать, что ошибка исправляется при декодировании.

ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ № 3

Для запуска необходимо открыть файл CODER.exe

1. Ввод исходных данных.

Разрядность кодера можно установить или вращением колеса мыши при наведении курсора на соответствующее поле, или один раз нажать ЛКМ и ввести с помощью клавиатуры.

Исходные данные

Разрядность K 3

вектор связи $g1$ 1 0 1

вектор связи $g2$ 1 1 0

сообщение m 1 1 0 1 1

Расчет

Отладочная информация:

$K=3$

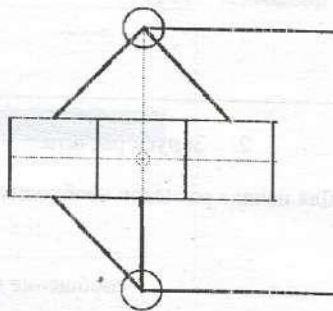
$g1=101$

$g2=110$

$m=11011$

Длина сообщения=5

Масштаб: 1



Ввод векторов связи и сообщения осуществляется только с помощью клавиатуры. Возможен ввод только чисел 1 или 0. Каждое число отделяется пробелом. Пример: 1 пробел 0 пробел 1.

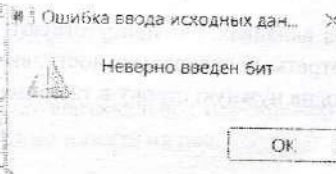
В случае ошибки ввода программа может выдать одно из следующих предупреждений:
если было введено число, отличное от 1 или 0

Разрядность K 3

вектор связи $g1$ 1 0 2

вектор связи $g2$ 1 1 0

сообщение m 1 1 0 1 1



длина верхнего и/или нижнего векторов отличается от заданной

Разрядность K	3
вектор связи g1	1 0 1
вектор связи g2	1 1 0
сообщение m	1 1 0 1 1
Расчет	

Ошибка ввода исходных данных

Несовпадение разрядности и длины вектора связи

ОК

2. Запуск расчета.

Для начала расчетов необходимо нажать кнопку «Расчет»

сообщение m	1 1 0 1 1
Расчет	

Отладочная информация:

3. Изменение масштаба.

При помощи ползунка под отладочной информацией можно изменять масштаб рисунка. В случае если масштаб не изменяется, следует повторно нажать кнопку «Расчет».

4. Вкладки.

На вкладках, где присутствуют таблица и схема кодера, возможно посмотреть последовательность вычислений. Для этого необходимо нажать на нужную строку в таблице.

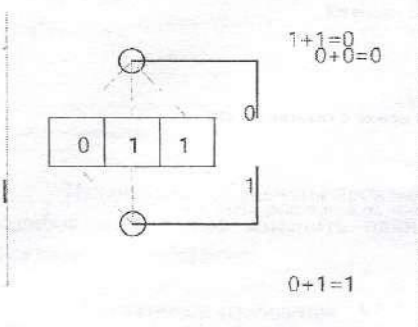
Исходные данные: реакция кодера на входное сообщение реакция на импульсное возмущение: 1

Входное сообщение: 1 1 1

	поступило	регистр	выход 1	выход 2	выход общий
1	1	100	1	1	11
2	1	110	0	0	00
3	0	011	0	1	01
4	1	101	0	1	01
5	1	110	0	0	00
6	0	011	0	1	01
7	0	001	1	0	10
8	0	000	0	0	00

11 00 01 01 00 01 10

Тестовая строка: 5



На вкладках, где присутствует вертикальное переключение, для открытия нужной вкладки необходимо нажать на нее.

таблица расчета и рисунок

Входное сообщение 1 0 0 0

поступило	регистр	выход 1	выход 2	выход общий
1 1	1 0 0	1	1	1 1
2 0	0 1 0	1	1	1 1
3 0	0 0 1	1	0	1 0
4 0	0 0 0	0	0	0 0

1 1 1 0

Текущая строка

расчет реакции на входное сообщение с помощью ИХ

таблица расчета и рисунок

расчет реакции на входное сообщение с помощью ИХ

Входное сообщение 1 1 0 1 1

Выходное сообщение, рассчитанное в предыдущем пункте: 1 1 0 0 0 1 0 1 0 0 1 0

ИХ: 1 1 1 1 0

1 1 1 1 0

1 1 1 1 0

0 0 0 0 0

1 1 1 1 0

1 1 1 1 0

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

ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ № 4

Диалоговое окно программы, описывающей работу декодера, выглядит следующим образом:

Входное сообщение: 11_00_01_01_00_01_10

Искаженное сообщение: 11_00_01_01_00_01_10

результат декодирования

таблица для приема без ошибок

вход	разница входа и реакции	регистр варианты	
1 00		000	
2 11	2/0	000/100 → 100	00/11 → 11
3 00	2/0	010/110 → 110	11/00 → 00
4 01	0/2	011/111 → 011	01/10 → 01
5 01	2/0	001/101 → 101	10/01 → 01
6 00	2/0	010/110 → 110	11/00 → 00
7 01	0/2	011/111 → 011	01/10 → 01
8 10	0/2	001/101 → 001	10/01 → 10
9 00	0/2	000/100 → 000	00/11 → 00
10 00	0/2	000/100 → 000	00/11 → 00

МАССЫ ПОИСКОВ ОШИБКИ

Изначально все расчеты производятся без ошибок. Для внесения ошибок необходимо изменить один или несколько бит в графе искаженное сообщение:

Искаженное сообщение: 11_00_01_01_00_01_10

Нажать кнопку «Расчет».

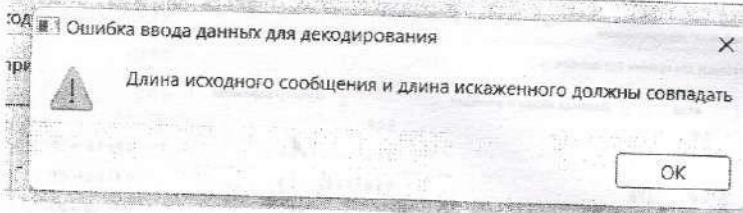
Биты задаются попарно через пробел. Каждая пара отделяется от другой нижним подчеркиванием:

Входное сообщение: 11_00_01_01_00_01_10

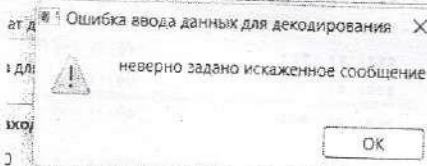
Искаженное сообщение: 11_00_01_01_00_01_10

В случае неправильного ввода будет выведено предупреждение:

сообщение 11_00_0101_00_01_10



исходное сообщение 11_00_01_01_00_01_10



Во вкладке «результат декодирования» выводится результат декодирования и количество возможных вариантов:

Искаженное сообщение 11_00_01_01_00_01_10

результат декодирования

Ошибка: 0

Состояния регистра: 000 100 110 011 101 110 011 001 000 000

Реакция кодера: 11 00 01 01 00 01 10 00 00

Результат: 0110110000

Во вкладке «таблица для приема без ошибок» можно отследить путь выбора решения на определенном шаге:

таблица для приема без ошибок

вход	разница входа и реакции	регистр варианты
1 00		000
2 11	2/0	000/100->100
3 00	2/0	010/110->110
4 01	0/2	011/111->011
5 01	2/0	001/101->101
6 00	2/0	010/110->110
7 01	0/2	011/111->011
8 10	0/2	001/101->001
9 00	0/2	000/100->000
10 00	0/2	000/100->000

Во вкладке «массив поиска ошибки» выводится таблица всех вариантов состояния декодера, а также указывается строка, в которой находится необходимое решение:

таблица для приема без ошибок

массив поиска ошибки

Строка 433

ошибка	состояние регистра	реакция декодера
430 5	000 100 110 011 101 010 101 110 011 101	11 00 01 01 11 01 00 01 01
431 5	000 100 110 011 101 010 101 110 111 011	11 00 01 01 11 01 00 10 01
432 5	000 100 110 011 101 010 101 110 111 111	11 00 01 01 11 01 00 10 10
433 1	000 100 110 011 101 110 011 001 000 000	11 00 01 01 00 01 10 00 00
434 2	000 100 110 011 101 110 011 001 000 100	11 00 01 01 00 01 10 00 11
435 4	000 100 110 011 101 110 011 001 100 010	11 00 01 01 00 01 10 11 11
436 2	000 100 110 011 101 110 011 001 100 110	11 00 01 01 00 01 10 11 00
437 5	000 100 110 011 101 110 011 101 010 001	11 00 01 01 00 01 01 11 10

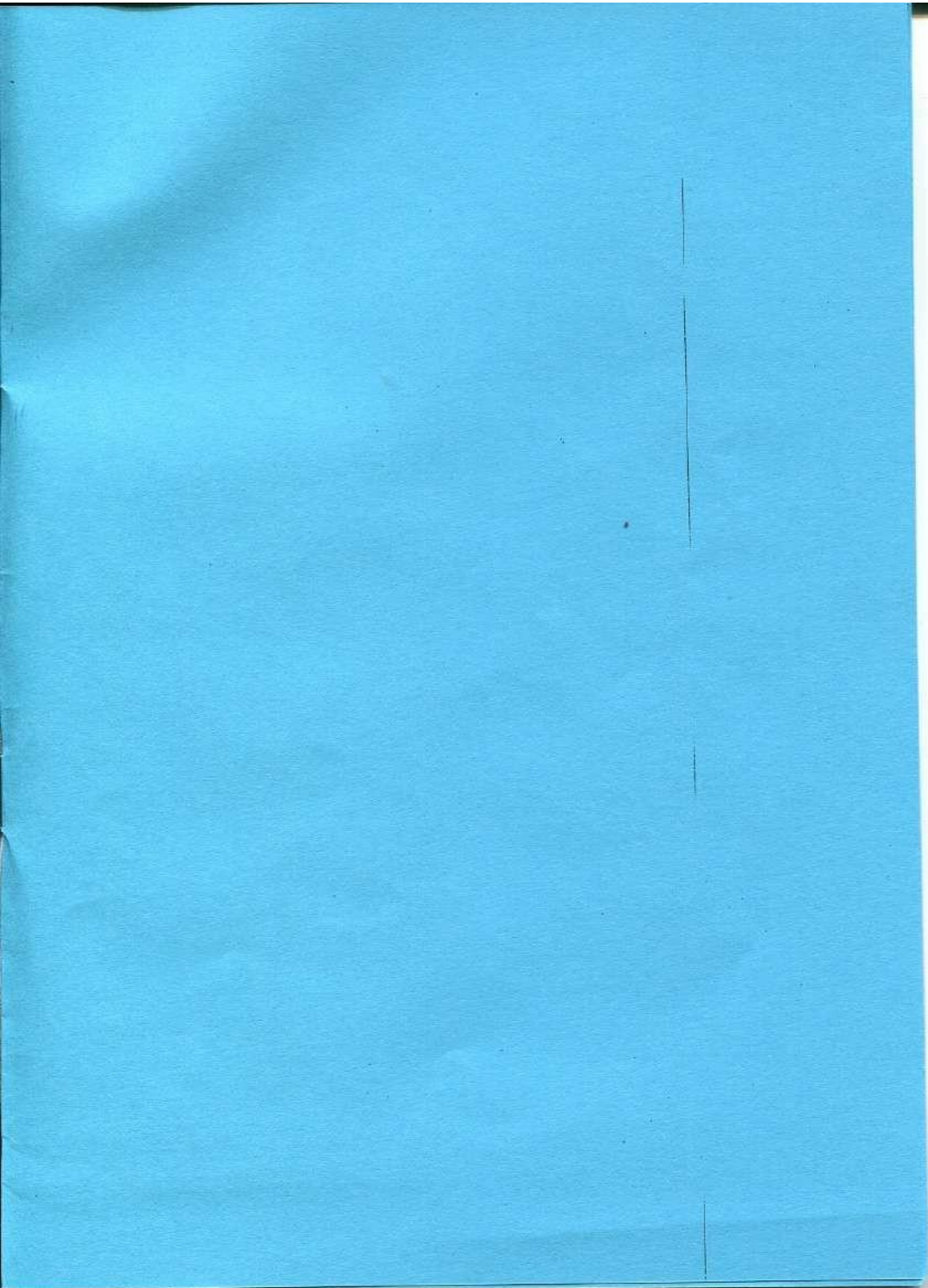
Нужная строка подсвечивается зеленым.

Контрольные вопросы

1. Структура сверточного кодера.
2. Реакция кодера на импульсное возмущение.
3. Полиномиальное представление.
4. Диаграмма состояний.
5. Решетчатая диаграмма.
6. Метод декодирования Витерби.

Библиографический список

1. Скляр Б. Цифровая связь. — М.: СПб.; Киев: Изд. дом «Вильямс», 2003.
2. Передача дискретных сообщений: учебник для вузов / В. П. Шувалов, Н. В. Захарченко, В. О. Шварцман и др.; под ред. В. П. Шувалова. — М.: Радио и связь, 1990. — 464 с.



Оператор ЭДО ООО "Компания "Тензор"

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ

СОГЛАСОВАНО

ФГБОУ ВО "РГРТУ", РГРТУ, Дмитриев Владимир
Тимурович, Заведующий кафедрой РУС

03.07.25 11:31 (MSK)

Простая подпись