Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет телекоммуникаций

Кафедра сетей и устройств телекоммуникаций

А. И. Королев, В. К. Конопелько

ТУРБОКОДЫ И ИТЕРАТИВНОЕ ДЕКОДИРОВАНИЕ

Рекомендовано УМО по образованию
в области информатики и радиоэлектроники
в качестве учебно-методического пособия
для специальности 1-45 81 01 «Инфокоммуникационные системы и сети»

Минск БГУИР 2015

УДК 621.391.7(076)
ББК 32.811.4я73
К68

Р е ц е н з е н т ы:

кафедра радиолокации и приемопередающих устройств
учреждения образования «Военная академия Республики Беларусь»
(протокол №1 от 08.09.2014);

главный научный сотрудник государственного учреждения
«Объединенный институт проблем информатики
Национальной академии наук Беларуси», доктор технических наук,
доцент С. Ф. Липницкий

Королев, А. И.

К68 Турбокоды и итеративное декодирование : учеб.-метод. пособие / А. И. Королев, В. К. Конопелько. – Минск : БГУИР, 2015. – 74 с. : ил.
ISBN 978-985-543-133-7.

Рассматриваются методы построения турбокодов, основные их параметры, алгоритмы декодирования, достоинства, недостатки и направления дальнейших исследований по повышению их эффективности.

УДК 621.391.7(076)
ББК 32.811.4я73

ISBN 978-985-543-133-7

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В связи с интенсивным развитием цифровых систем передачи и обработки информации актуальной задачей является обеспечение высокой ее достоверности или минимальной вероятности ошибочного приема. Эффективным способом решения этой задачи является применение помехоустойчивого или корректирующего кодирования информации. Следовательно, выбор помехоустойчивого кода, метода кодирования и алгоритма декодирования информации является актуальной задачей теории и практики помехоустойчивого кодирования.

В соответствии с доказательствами К. Шеннона наилучшим кодом является код, который передает сообщение за бесконечно большое время, формируя в каждый момент времени случайные кодовые элементы. У приемника есть бесконечные версии сообщения, искаженного случайным образом. Из этих копий декодер должен выбрать копию, наиболее близкую к переданному сообщению. Это представляет собой теоретический идеальный код, который может исправить все ошибки в сообщении.

Как отмечал К. Шеннон, можно легко найти много «хороших» кодов, но очень трудно найти простые алгоритмы их декодирования [1]. Первым шагом в этом направлении было открытие в 1964 г. каскадных кодов, а вторым более успешным шагом в этом направлении было открытие в 1993 г. К. Берроу, А. Главье и П. Ситимашимой [2, 3] турбокодов. В целом турбокоды (ТК) относят к случайным кодам.

Турбокод – параллельный каскадный либо блоковый, либо сверточный код, связанный перемежителем информационных символов, способный исправлять ошибки, возникающие при передаче цифровой информации по каналу связи с шумами (помехами). Составляющие коды называются *компонентными.*

Термин «турбо» отражает свойства, используемые при декодировании *итеративного алгоритма*: информация с выхода одной итерации (этапа, процедуры) декодирования поступает на вход второй итерации декодирования и т. д. Количество итераций декодирования зависит от обеспечения заданной достоверности передачи информации и

может составлять несколько десятков. Для декодирования составляющих кодов обычно используют декодирование по *максимуму апостериорной вероятности* (MAP-алгоритм) или его упрощенную версию MLM (Max-LogMAP). В связи с этим ТК позволяют практически приблизиться к так называемой границе Шеннона: реальный проигрыш составляет не более 0,5 дБ. Благодаря этому свойству ТК находят в настоящее время широкое применение в цифровых системах связи различного назначения и применения.

1. КОДИРУЮЩИЕ УСТРОЙСТВА ТУРБОКОДОВ: ПРИНЦИПЫ ПОСТРОЕНИЯ И ХАРАКТЕРИСТИКИ

1.1. Общий принцип построения кодирующих устройств турбокодов

В соответствии с [1–5] кодирование информационных символов передаваемых сообщений осуществляется турбокодированием на основе использования N(N≥2) кодирующих устройств (кодеров), работающих параллельно с реализацией дополнительной процедуры – перемежения информационных символов во 2-м, 3-м и последующих каналах кодирования. На рис. 1.1 приведена обобщенная структурная схема канального кодера турбокода при использовании N = 2 каналов кодирования. В целом число каналов кодирования декодирования может быть больше двух.

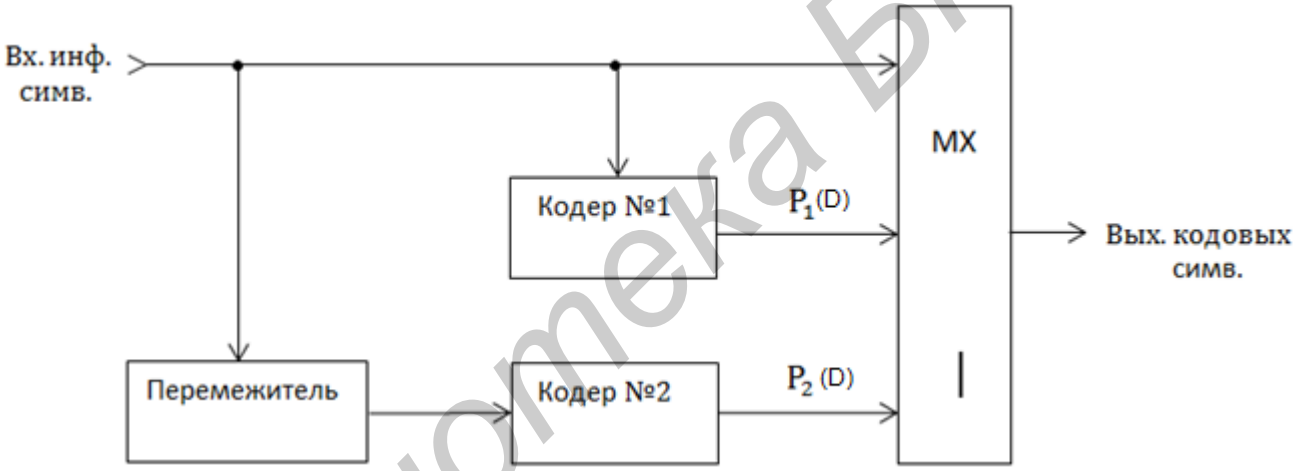


Рис. 1.1. Обобщенная структурная схема канального кодера ТК при N = 2 каналах кодирования

Канальный кодер ТК работает следующим образом. Передаваемые информационные символы сообщения поступают одновременно на первый вход мультиплексора (МХ), на вход кодера №1 первого канала кодирования и через перемежитель информационных символов на вход кодера №2 второго канала кодирования. Сформированные проверочные (контрольные) символы первого (P_1) и второго (P_2) каналов кодирования поступают соответственно на второй и третий входы МХ, который формирует последовательный поток кодовых символов n'. Передача кодовых символов может производиться как в

систематическом коде ($n^{'}$ = $ip_1p_2ip_1p_2$…), так и вне систематического кода ($n^{'}$ = $p_1ip_2ip_1p_2i$…); на практике наибольшее применение получил способ передачи кодовых символов в систематическом коде. Выбор типа помехоустойчивого кода и принципа построения кодеров каналов кодирования является ответственным моментом в формировании ТК и построения канального кодера.

Турбокодеры могут выполняться на основе использования *сверточных кодов, блочных кодов и кодов с низкой плотностью проверок на четность.* Однако на практике наибольшее применение получили следующие схемы турбокодов [4–9]:

– параллельное включение рекурсивных сверточных кодов;

– последовательное включение рекурсивных сверточных кодов;

– турбокоды на основе использования блочных кодов; чаще всего используются коды Хэмминга, соединенные последовательно без промежуточного перемежителя.

Из [1–9] известно, что корректирующая способность помехоустойчивых кодов определяется *минимальным кодовым расстоянием* d_0 (d_x – *хэмминговым расстоянием* между кодовыми словами (КС) или кодовыми последовательностями (КП)), часто называемым структурой кода. Для сверточного кода структура кода определяется длиной кодового ограничения, равного v = k +1, где k – количество ячеек памяти регистра сдвига кодера. При использовании ТК структура кода определяется как свойствами кодера, так и свойствами перемежителя информационных символов. При разработке новых кодов в качестве критерия оценки их эффективности используют критерий – максимум *минимального кодового расстояния* $d_{0\max}$ между кодовыми словами. Однако коды с большими значениями $d_{0\max}$ имеют большую сложность реализации. Эффективность ТК определяется в основном не максимальным значением $d_{0\max}$, а большей частью средним значением расстояний $d_{0\text{ср}}$ между кодовыми словами, так как в процедуре кодирования используется перемежитель информационных символов. В соответствии с этим при формировании выходного кодового слова или КП из двух практически независимых кодовых слов или двух КП значение $d_{0ср}$ их суммы будет значительно больше, чем d_0 исходного кода [10].

Корректирующая способность ТК, в отличие от корректирующей способности СК, существенно зависит от распределения зависимых слов, расположенных на расстоянии d от других кодовых слов, т. е. от вида *функции*

распределения S(d), описывающей спектр кодовых расстояний и в большей степени от той ее части, где d_0 меньше d_{0cp} [19]. Это значит, что в отличие от других кодов, где для оценки их эффективности достаточно знать лишь некоторые параметры функции распределения S(d), для более точной или корректной оценки эффективности ТК необходимо иметь достаточно полное описание этой функции. Свойства *функции распределенияS(d)* для ТК зависят как от скорости и типа кода, так и от параметров перемежителя информационных кодов. Установлено [1–10], что эффективность ТК увеличивается с увеличением длины кодового ограничения СК и длины интервала перемежения информационных символов и слабо зависит от структуры перемежителя.

В реальных системах передачи информации широкое применение получили турбодекодеры, реализуемые на основе *рекурсивных сверточных кодов. Рекурсивный сверточный код* (RSC-PCC) – это разновидность сверточного кода (СК), в котором входные информационные символы передаются непосредственно на выход кодера, а проверочные (избыточные) символы формируются (генерируются) логической цепью, содержащей регистр сдвига с обратной связью. Установлено [1–17], что применение двух параллельных RSC-кодеров ТК с перемежителем информационных символов перед вторым из них приводит к формированию (генерированию) кодов с «очень хорошими свойствами», т. е. к формированию кодов с бóльшими кодовыми расстояниями, чем исходные коды.

На рис. 1.2 приведена обобщенная функциональная схема турбокодера на основе использования двух RSC-кодеров.

Оба RSC-кодера работают со скоростью кода R = 1/2. Это значит, что на входе RSC-кодера на один информационный символ кодер выдает два кодовых символа. Значение символа x на систематическом выходе верхнего RSC-кодера совпадает со значением входного символа, а на втором выходе RSC-кодера формируется проверочный символ y_1. На вход второго RSC-кодера с выхода перемежителя поступает информационный символ, номер которого j зависит от номера i на входе перемежителя по псевдослучайному закону (i, j = 1, …, K). Для этого блок из K информационных символов предварительно перед операцией кодирования должен быть записан в память. У второго RSC-кодера систематический выход x′ не используется, а на втором выходе формируется второй проверочный сигнал y_2. С выхода канального (общего) турбокодера на вход модулятора сначала поступает информационный символ x, а затем последовательно поступают два проверочных символа: y_1 – с первого RSC-

кодера и y_2 – со второго RSC-кодера. В результате скорость кода канального турбокодера становится равной R=1/3, т. е. относительная избыточность ТК составляет

$$R = (1 - R)\cdot 100\ \% = (1 - 0{,}33)\cdot 100\ \% = 67\ \%.$$

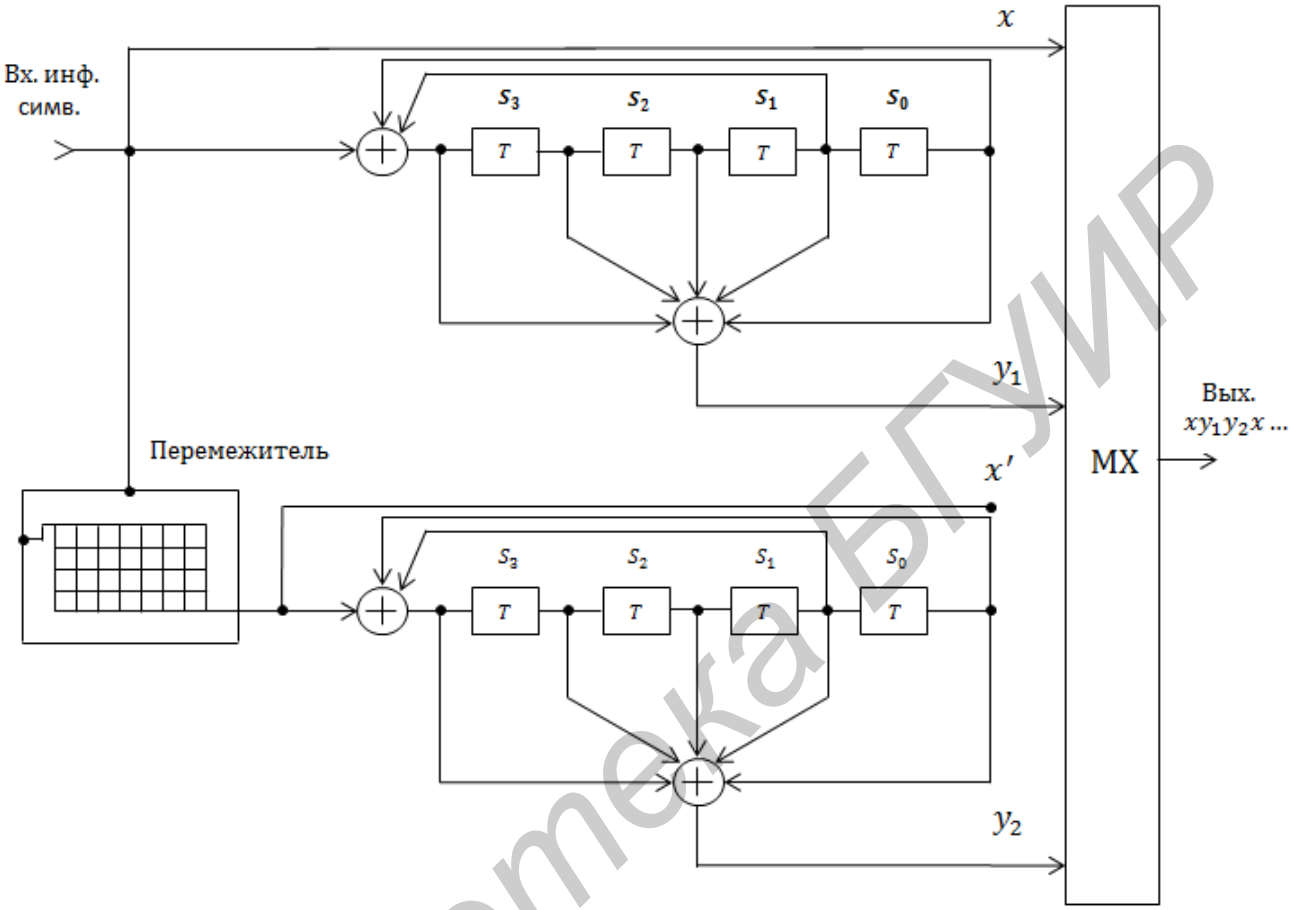


Рис. 1.2. Обобщенная функциональная схема турбокодера на основе двух RSC

Свойства RSC-кодеров зависят от структуры СК и могут быть определены по его диаграмме состояний. На рис. 1.3 представлена диаграмма состояний RSC первого канала кодирования.

Так как регистр сдвига RSC-кодера имеет четыре ячейки памяти, то диаграмма RSC имеет 16 состояний. Адреса состояний – десятичное значение двоичной последовательности $S_{0\text{-}3}$. Стрелки, связывающие каждое состояние, представляют символ x_k^s в RSC-кодере в промежутке времени k: сплошная стрелка представляет собой входной нулевой символ, а пунктирная стрелка – входной ненулевой символ. Кроме того, стрелка «0» или «1» представляет символ контроля четности x_k^s, связанный с каждым переходом. Связь между

состояниями зависит от обратной связи RSC-кодера, а символы контроля четности, связанные с каждым переходом, зависят от величины части сдвига.

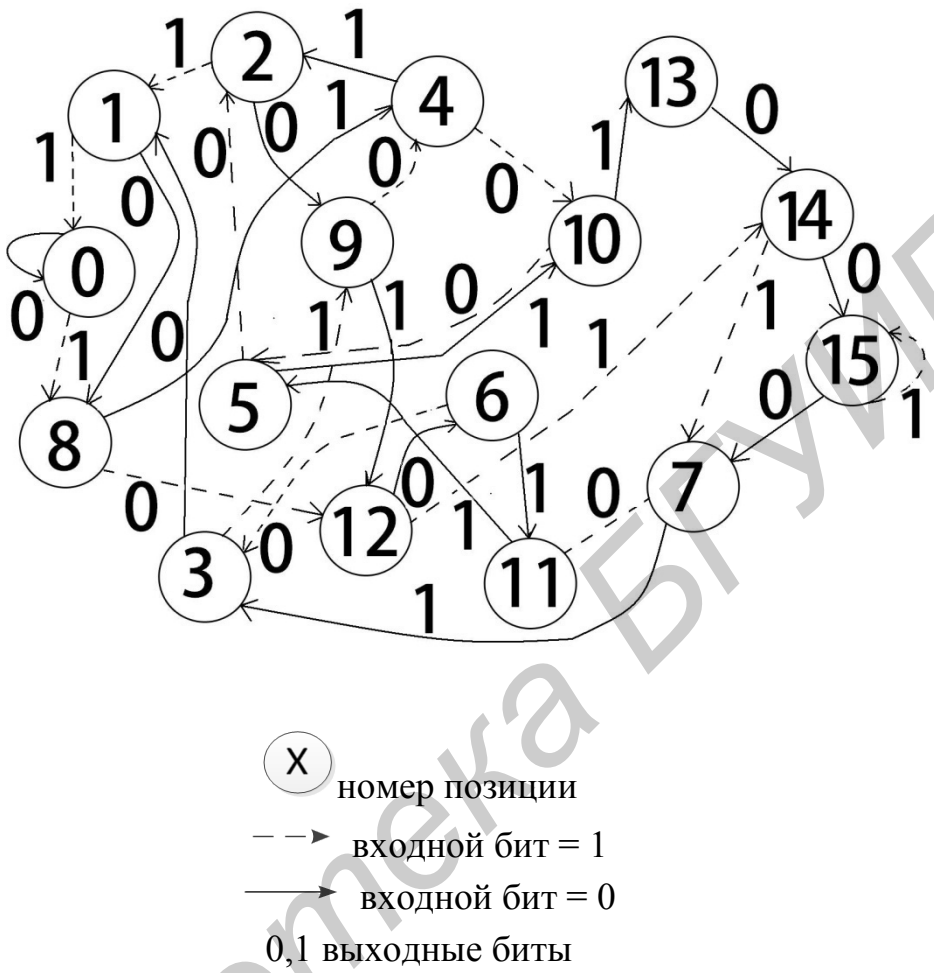


Рис. 1.3. Диаграмма состояний RSC-кодера с порождающим полиномом g(D) = {23;37}

Диаграмма состояний RSC-кодера показывает, что если передавали нулевую последовательность символов, то RSC-кодер как конечный автомат будет находиться в состоянии «D»: ненулевой символ данных, представленный как $x_k^s = 1$ (пунктирная линия), переключит конечный автомат в состояние 8. Последующий нулевой символ данных, представленный как $x_k^s = 0$ (сплошная линия), будет переключать конечный автомат во всех остальных состояниях для данного порождающего полинома, формируя (накапливая) паритетный вес (parityweight) или общее число нулевых символов «1» в кодовом потоке данного RSC-кодера, когда символ четности для перехода автомата равен ненулевому символу. Так как RSC-кодер реализуется на основе RSC

(рекурсивных систематических кодов), то RSC-кодер как конечный автомат будет продолжать работу до бесконечности, накапливая паритетный вес, пока символы данных на входе RSC-кодера (как первого, так и второго каналов кодирования) остаются нулевыми. Однако, если через 15 отсчетов времени после появления первого нулевого символа появится еще один нулевой символ данных, то конечный автомат будет переведен в нулевое состояние. Это означает, что работа конечного автомата закончена или прервана. Последующий нулевой символ данных будет сохранять конечный автомат данных в состоянии «0» и не формировать (генерировать) дальнейший паритетный вес.

Вес RSC будет равен 10 – общее количество символов четности, равных единице. Если пройти все состояния автомата, начиная и заканчивая в состоянии «0», пары ненулевых символов данных, размещенные на позициях, кратных 15, также вернут RSC-кодеру нулевое состояние.

В общем случае известно, что вероятность ошибки, происходящая в каждом узле диаграммы, для всех каналов (путей), которые отклонились от нулевого канала (пути), определяется весом Хэмминга или весом кода. Особое внимание уделяется ненулевым символам данных, которые расположены в таком интервале, что декодер возвращается обратно в нулевое состояние после короткого периода времени. Такие явления декодирования называются *самоограниченными последовательностями* и, как правило, имеют низкий вес кода.

Если *самоограниченная последовательность* любого типа, т. е. структуры, затем будет перемещена перемежителем в другую последовательность, которая не является *самоограниченной*, то второй RSC-кодер может зацикливаться вокруг ненулевого состояния много раз, накапливая большой паритетный вес (рис. 1.4).

Вес кода результирующей последовательности для скорости передачи кода R = 1/3 (без перфорации или выкалывания кода) определяется по формуле

Вес кода = вес данных + паритетный вес RSC_1 + паритетный вес RSC_2.

Этот вес будет стремиться к высокому значению (уровню). Расчет вероятности ошибочного декодирования, выполняемый по формуле [23]

$$P_e \leq \sum_{d=d_0} \alpha(d) a\left(\sqrt{2dR\frac{E_в}{N_0}}\right), \qquad (1.1)$$

показывает, что высокий вес кода между двумя кодовыми словами приводит к малой вероятности парной ошибки.

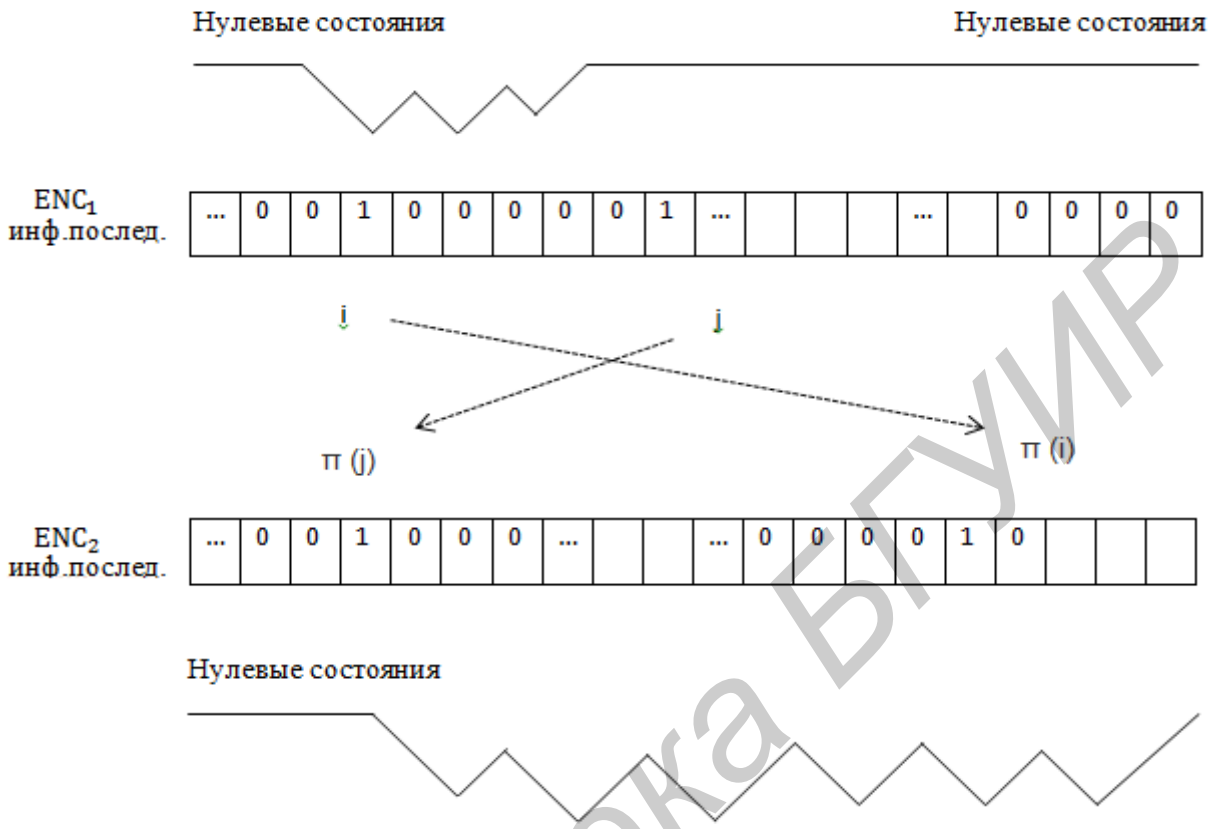


Рис. 1.4. Преобразование перемеженных символов, соответствующих «хорошему» распространению ненулевых символов данных

Однако высокий вес в некоторых случаях перемежитель информационных символов может заменить *самоограниченную последовательность* для RSC_1 на *самоограниченную последовательность* RSC_2. Результирующее кодовое слово было названо общим конечным кодовым словом (globalfinitecodeword) [23]. Такая ситуация представлена на рис. 1.5.

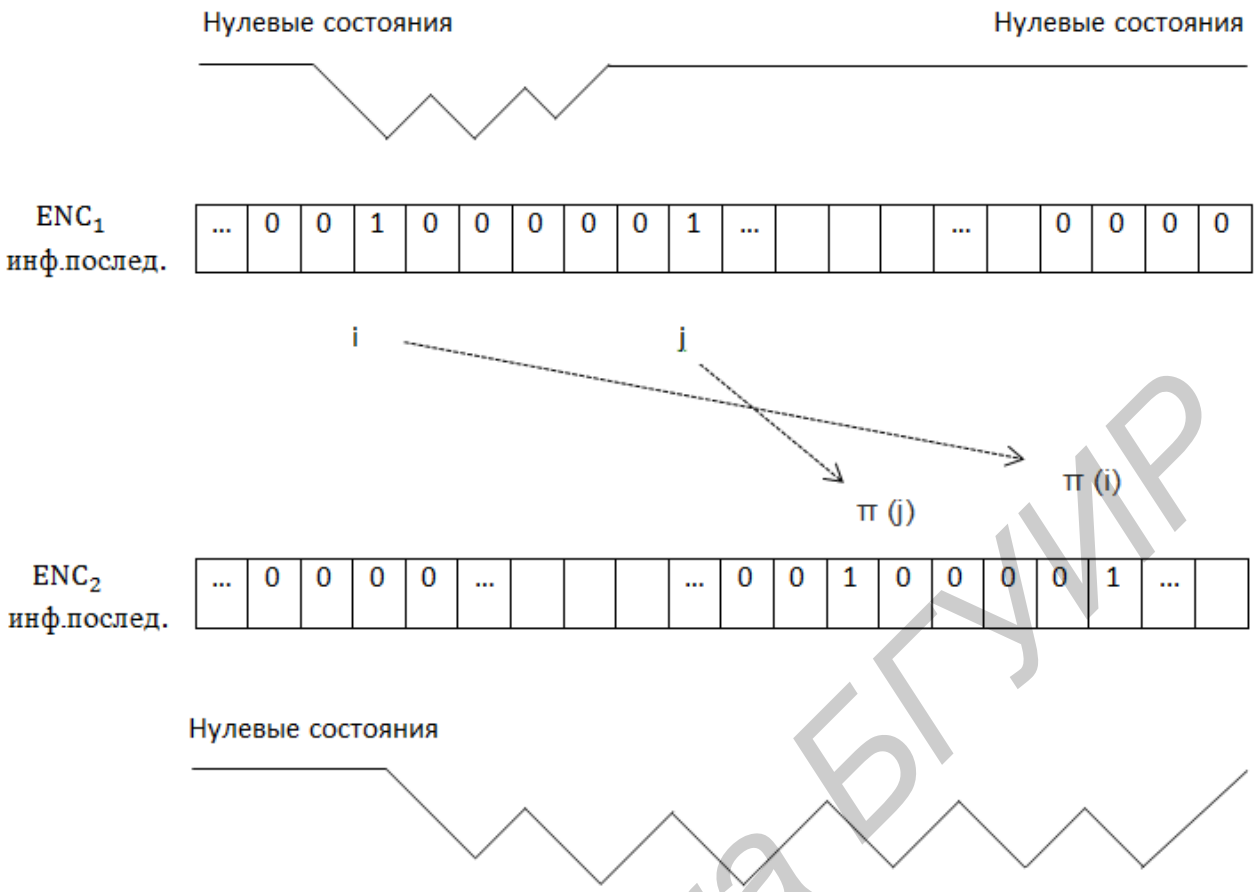


Рис. 1.5. Преобразование перемеженных информационных символов с низким весом кодовой последовательности для обоих RSC

Оба RSC-кодера будут возвращены в нулевое состояние и не будут иметь возможности накапливать паритетный вес. Например, если снова принять во внимание диаграмму состояний, а затем, если пара нулевых символов данных, расположенных на 15 символах (узлах), будет перемещена так, что они по-прежнему будут расположены на 15 позициях друг от друга, то результирующий вес кода будет равен

Вес кода = вес данных + паритетный вес RSC_1 +паритетный вес RSC_2 =

= 2 + 10 + 10 = 22.

Это существенно меньше, чем ожидаемый кодовый вес большинства парных преобразований в перемежителе. Такой низкий вес кода приводит к более высокой вероятности ошибки на коде (бит). Высокая кратность α(d) таких последовательностей в перемежителе только увеличивает сложность декодирования. Из диаграммы состояния видно, что данная проблема не ограничивается парами символов данных. Оказывается, что есть много других

последовательностей, например […0100110…], которые являются самоограниченными. Установлено, что имеется большая вероятность того, что при использовании случайного перемежения информационных символов эти три символа (бита) в большинстве случаев или в большей степени могут влиять на кодовое слово, чтобы оно не считалось самоограниченным.

Таким образом, можно отметить, что роль перемежителя информационных символов состоит в том, чтобы обеспечить переход большей части низковесных самоограниченных последовательностей одного компонента RSC (или ЦК) внесамоограниченную последовательность с большим весом другого компонентного RSC (или ЦК).

Представленный на рис. 1.2 канальный турбокодер обеспечивает передачу кодовых символов со скоростью кода R = 1/3 и избыточностью, равной 67 %, что является существенным недостатком. Повысить эффективность турбокода можно, если использовать перфорированный или выколотый (puncturing) код.

1.2. Общий принцип перфорирования сверточных кодов

Сущность перфорации сверточных кодов и, в частности, рекурсивных сверточных кодов (RSC) состоит в том, что из общей последовательности сформированных кодовых символов ТК по определенному правилу исключается на передачу в канал связи часть проверочных символов компонентных RSC. Перфорирование или выкалывание проверочных символов кодового слова или кодовой последовательности ТК позволяет формировать ТК с разной кодовой скоростью передачи и корректирующей способностью. Однако следует учитывать, что одна процедура (один этап) перфорации приводит к энергетическому проигрышу помехоустойчивого кодирования примерно на 0,2 дБ. На рис. 1.6 приведены примеры получения(формирования) перфорированных ТК при использовании двухкомпонентных RSC с R = ½.

Кодовые символы, которые передаются в канал связи, обозначены числом 1, а которые исключаются из передачи – числом 0. В правой части рис. 1.6 представлены кодовые символы, которые передаются в канал связи. В случае «а» ТК не перфорирован и его кодовая скорость R = $^1/_3$, а в случае «б» чередующиеся проверочные символы от каждого из компонентных RSC исключаются из передачи в канал связи на каждом промежутке времени k. В результате кодовая скорость ТК равна R= ½. В случае «в» исключается

большее число проверочных символов и в результате кодовая скорость ТК равна R = ¾. На рис. 1.7 приведен пример перфорации компонентного RSC при кодировании данных и восстановления исключенных проверочных символов при организации декодирования RSC.

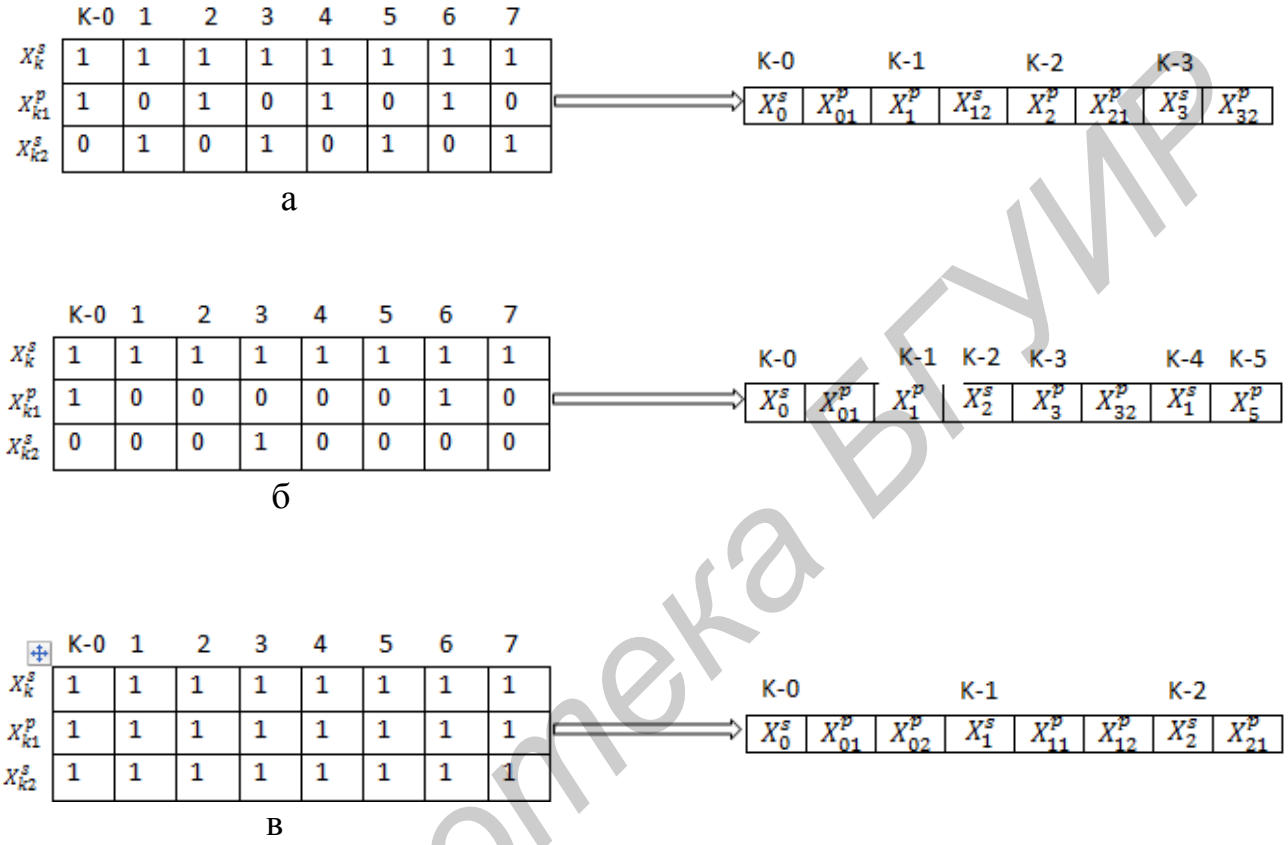


Рис. 1.6. Примеры формирования перфорированных турбокодов:
а – турбокод со скоростью ⅓; б – турбокод со скоростью ½ ;
в – турбокод со скоростью ¾

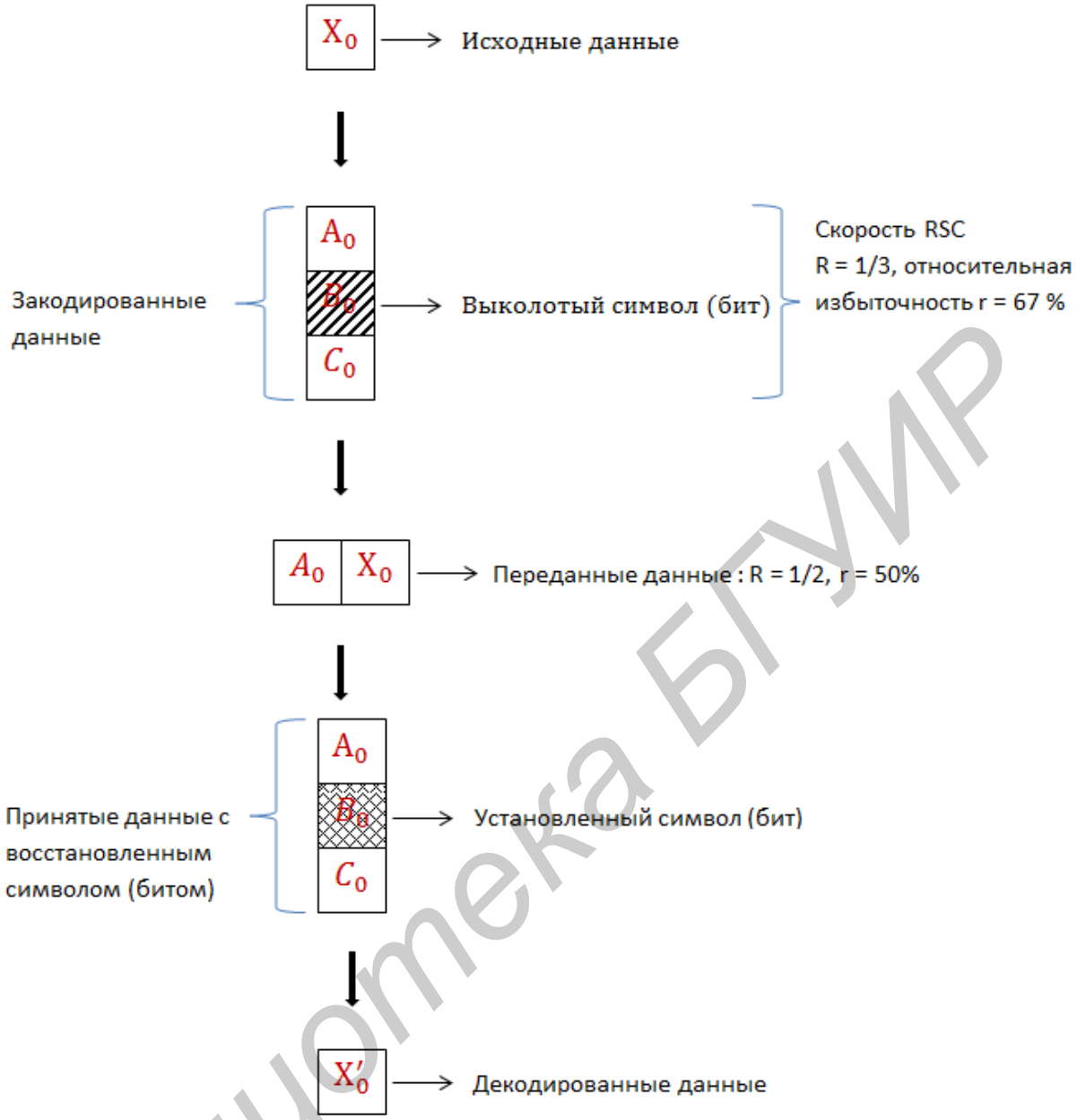


Рис. 1.7. Пример формирования перфорированного и восстановленного RSC (R = ½)

Для случая «в» (см. рис. 1.6) пример перфорирования и восстановления кодовых символов RSC приведен на рис. 1.8.

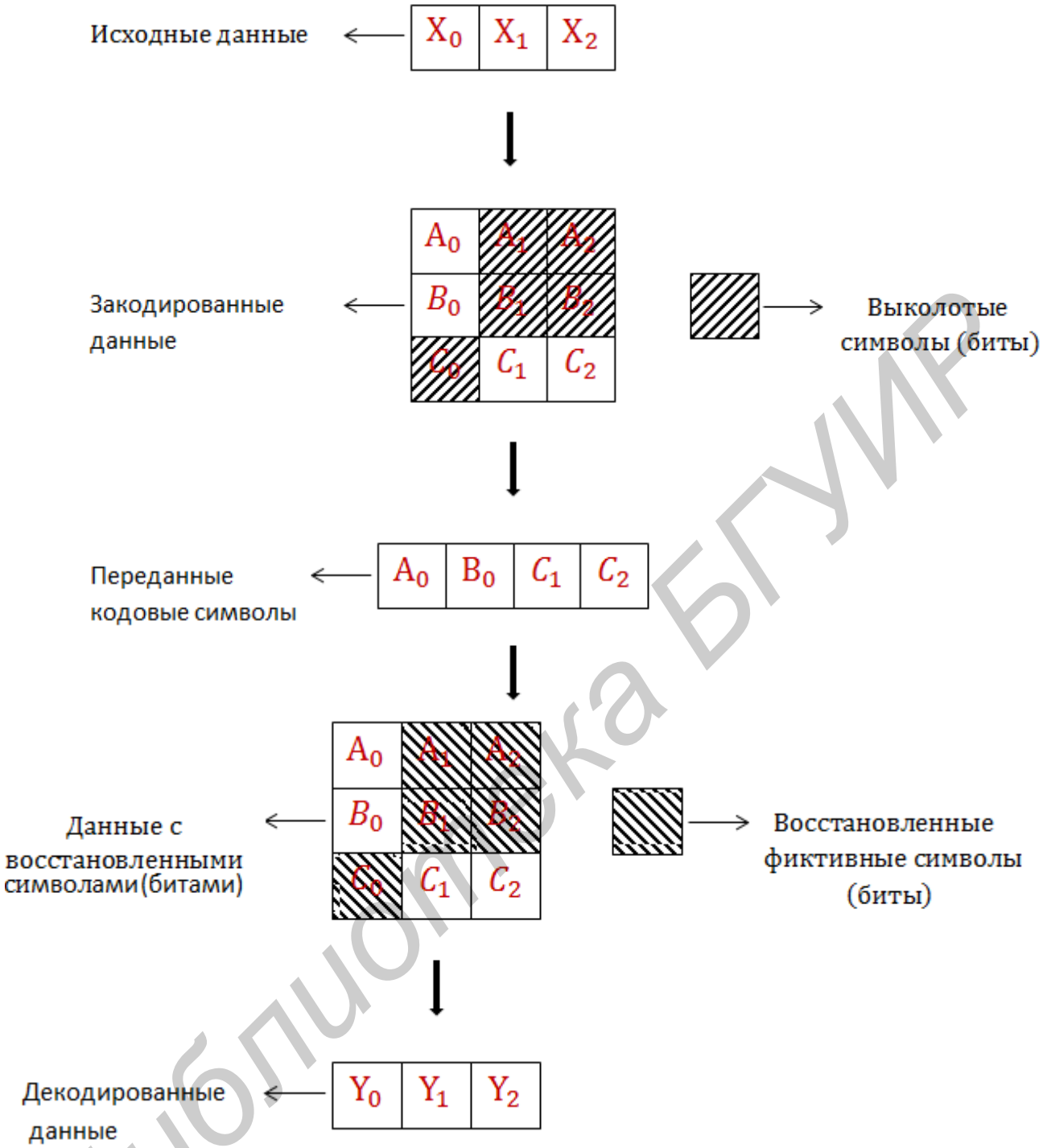


Рис. 1.8. Пример формирования перфорированного и восстановленного RSC (R = ¾)

1.3. Процесс рандомизации (завершения) процедуры кодирования данных сверточными кодами

Принципиальное отличие сверточных кодов (СК), в том числе и RSC, состоит в том, что сверточное кодирование данных представляет собой непрерывный процесс. То есть СК не имеют, как блоковые коды, кодовые слова или кодовые последовательности определенной длины. Однако ТК имеют определенную или фиксированную блоковую длину, которая определяется длиной интервала перемежения перемежителя.

Следовательно, при организации декодирования RSC-кодеров необходимо, чтобы были обнулены или рандомизированы регистры сдвига обоих RSC-кодеров. С этой целью при кодировании к каждому блоку информационных символов добавляются нулевые символы, названные «остаточными символами», а сама процедура вставки нулевых символов носит название «Завершение» процесса кодирования. Данная процедура кодирования/декодирования позволяет алгоритму декодирования RSC правильно принимать решения о начальном и конечном состояниях кодовой решетки. Завершение или рандомизация обоих RSC-кодеров является более сложным процессом, чем для одного RSC-кодера. Это определяется тем, что завершающая (нулевая) последовательность для первого RSC-кодера перемежена и второй RSC-кодер может не завершить самостоятельно процесс рандомизации. Устройство перемежения и деперемежения должно быть выполнено таким образом, чтобы осуществлялось перемежение завершающей последовательности первого RSC-кодера в завершающей последовательности второго RSC-кодера. В этом случае погрешность вероятности ошибочного декодирования ТК будет наименьшей и существенно меньше, чем при процессе завершения для одного из компонентных RSC-кодеров.

1.4. Общий принцип построения декодирующих устройств турбокодов

Из приведенных структурной и функциональной схем канальных кодеров ТК (см. рис. 1.1 и 1.2) следует, что канальный декодер ТК может быть выполнен в виде каскадного соединения отдельных индивидуальных RSC-декодеров, количество которых зависит от числа используемых итераций декодирования [1–9]. На рис. 1.9 и 1.10 приведены обобщенные структурные

схемы канальных декодеров ТК при использовании соответственно одной и трех итераций декодирования.

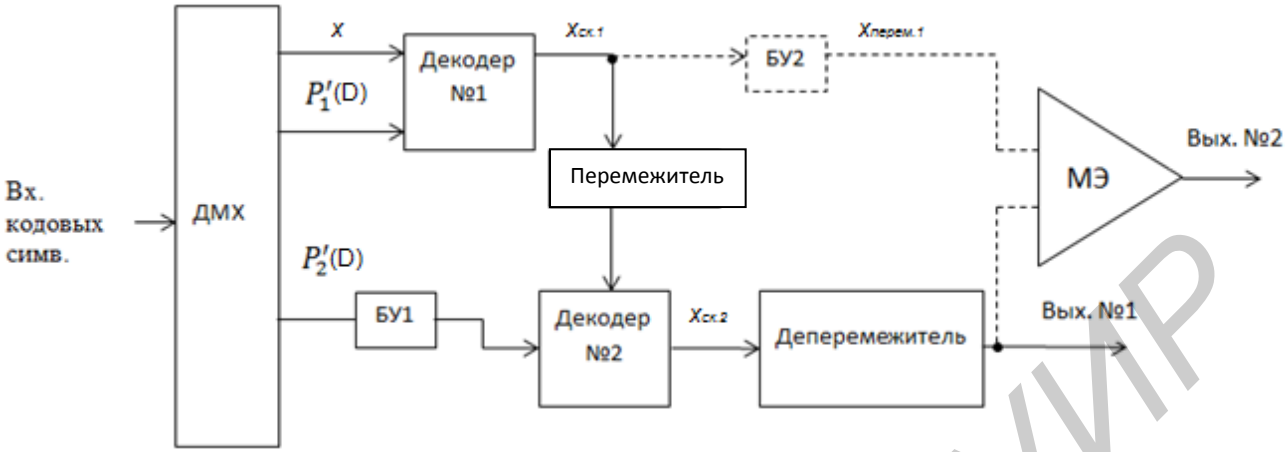


Рис. 1.9. Канальный турбодекодер с одной итерацией декодирования (Q = 1)

Принятые кодовые символы в ДМХ (демультиплексоре) разделяются на три канала или на три потока кодовых символов: x’ – поток неперемеженных информационных символов; $P_1(D)$ – поток проверочных символов, сформированных из неперемеженных информационных символов первым RSC-кодером канального кодера; $P_2’(D)$ – поток проверочных символов, сформированных из перемеженных информационных символов. Далее возможны различные варианты организации декодирования кодовых символов. В данном варианте построения канального декодера ТК с жестким принятием решения на выходе демодулятора ДКС первоначально осуществляется декодирование неперемеженных информационных символов на основе информации x’ и $P_1’(D)$.Скорректированные на первом этапе информационные символы $x_{ск.1}$ поступают на вход перемежителя информационных символов, имеющих конструкцию, аналогичную перемежителю канального кодера. Скорректированные перемеженные информационные символы поступают на вход второго декодера, на второй вход которого поступают проверочные символы $P_2’(D)$, сформированные вторым RSC-кодером канального турбо-кодера. Далее осуществляется второй этап декодирования. Подобные операции декодирования можно производить многократно и в этом состоит сущность турбо- или итеративного декодирования (рис. 1.10).

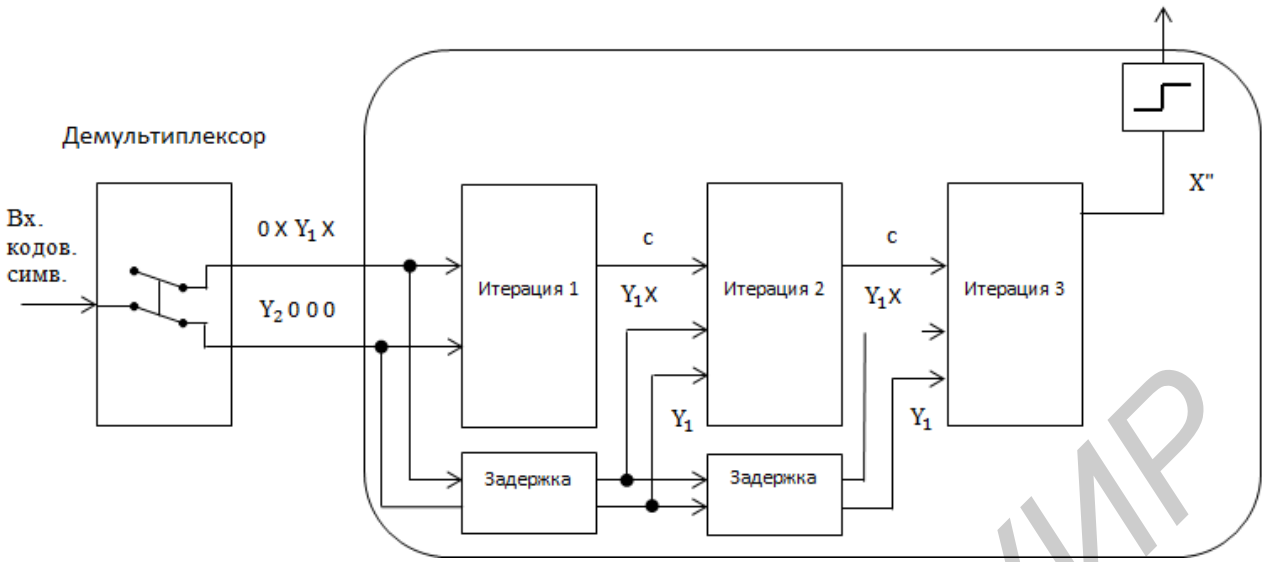


Рис. 1.10. Канальный турбодекодер с тремя итерациями декодирования (Q = 3)

Скорректированные на втором этапе декодирования информационные символы $x_{ск.2}$ могут выдаваться получателю информации, как это показано пунктиром на рис. 1.8, или поступать на один из входов мажоритарного элемента (МЭ), на второй вход которого поступают информационные символы, скорректированные на первом этапе декодирования. Данная дополнительная операция декодирования позволяет повысить достоверность принятия решения о полярности декодируемого информационного символа. Приведенные операции итеративного декодирования являются лишь приближенным описанием алгоритма функционирования итеративного декодера или турбодекодера. В общем оптимальный декодер должен быть построен на основе критерия минимума вероятности ошибочного декодирования. Однако построение такого декодера из-за наличия перемежителя является трудной задачей, и построение декодеров, реализующих подоптимальные алгоритмы декодирования, является актуальным мероприятием.

Следует отметить, что назначение итеративного декодирования состоит не только в том, чтобы обеспечить минимум вероятности ошибочного декодирования путем нахождения более оптимальной формы закона распределения *функций* S(α), а также в том, чтобы сформировать критерий достоверности декодирования путем сочетания структуры (конструкции) псевдослучайного перемежителя со структурой используемых кодов, и в случае

формирования «плохого» первого кодового слова (блока), т. е. близко расположенного от соседних кодовых слов (блоков), стремиться сделать второе кодовое слово (блок) «хорошим», т. е. расположить кодовое слово далеко от соседних кодовых слов. В этом случае в целом повышается достоверность декодирования информационных символов и сообщений в целом. Решению этой задачи посвящено значительное число работ по ТК.

Вариант построения канального декодера ТК, представленного на рис. 1.9, предусматривает реализацию итеративного декодирования RSC при «мягком» принятии решения на выходе ДКС. На выходе первого RSC-декодера формируется оценка («мягкое» решение) информационного символа, которое затем используется в качестве априорной информации о нем для второго RSC-декодера. Второй RSC-декодер производит оценку информационного символа с выхода перемежителя на основе проверочной части второго кодового слова. На второй итерации эта оценка обновляется и используется как априорная информация о переданном символе для первого RSC-декодера. Таким образом, на вход каждого из двух индивидуальных RSC-декодеров поступают *мягкие решения* и результаты декодирования на их выходах также представляются в «мягком» виде. По этой причине такие алгоритмы декодирования помехоустойчивых кодов получили название – декодирование с «мягким» входом и «мягким» выходом (SISO – SoftInputSoftOutput) [1–9].

Окончательное принятие решения о переданных информационных символах выносится третьим RSC-декодером (см. рис. 1.10). В целом окончание процесса декодирования происходит либо после выполнения заданного количества итераций (итерационных циклов – Q), либо после достижения допустимой вероятности ошибочного декодирования. Минимальная вероятность декодирования ТК существенно зависит от длины интервала перемежения, количества используемых итерационных циклов и алгоритмов декодирования, реализуемых индивидуальными RSC или блоковыми декодерами канального декодера ТК.

1.5. Алгоритмы турбодекодирования

В канальных декодерах ТК наибольшее применение получили следующие вероятные алгоритмы декодирования помехоустойчивых кодов [1–5]:

– MAP (Maximum of A-posteriori Propability) – максимум апостериорной вероятности;

– Log-MAP – логарифмический максимум апостериорной вероятности;

– Max-Log-MAP – максимальный логарифмический максимум апостериорной вероятности;

– SOVA (SoftOutputViterbiAlgorithm) – алгоритм декодирования Витерби.

Сущность данных алгоритмов декодирования подробно рассматривается в соответствующей литературе [13–18]. Далее описывается только общая сущность данных алгоритмов.

1.5.1. Алгоритм Витерби

В соответствии с [1, 8, 14] сущность алгоритма Витерби состоит в следующем:

1. Для каждого кодового мини-блока длиной n_0, соответствующего числу k_0 входных информационных символов, вычисляются *метрики* (оценки или веса) ветвей (ребер) по алгоритму

$$d_u = \omega\big(C(D) + C'(D)\big),$$

где d_u – метрика (расстояние, оценка, вес) для двух кодовых последовательностей;

C(D) – символы принятых кодовых последовательностей $T_1(D)$ и $T_2(D)$;

C'(D) – символы разрешенных кодовых последовательностей $T_1(D)$ и $T_2(D)$.

Данный алгоритм должен выполняться декодером на каждом такте для вновь поступивших n_0 кодовых символов. Таким образом, *метрика ребра (ветви)* решетчатой диаграммы – это путь каждого шага декодирования.

2. Вычисляются метрики всех путей, ведущих к данному углу (вершине) решетчатой диаграммы, для чего к метрике, которая была вычислена в предыдущем такте, прибавляется новая метрика, вычисленная в новом такте, т. е. для вновь поступившего на вход декодера мини-блока n_0, по алгоритму

$$d_{ij} = d_{j(i-1)} + d_i.$$

На данном этапе декодирования отбрасываются все пути с большими значениями d. Это обеспечивает уменьшение объема вычислений и соответственно повышение быстродействия обработки информации декодером.

3. Для каждого узла решетчатой диаграммы отобрать кодовый путь с наименьшим весом (расстоянием) и сохранить его. Данный путь считается «выжившим» путем.

4. Приписать узлу вес (расстояние) «выжившего» пути.

5. Записать в «память путей» выжившие пути.

6. По окончании вычислений в качестве решения выдается декодером СК тот путь (информационная последовательность), который имеет наименьший вес (расстояние).

Метрика (вес, расстояние) всего пути, соответствующая выдаваемой информационной последовательности, складывается из метрик, получаемых на отдельных тактах. В конце вычислений в качестве решения, как указывалось выше, будет выбран путь с наименьшим расстоянием $d_{\sum ij}$.

Блок-схема, поясняющая АВ, может быть представлена в виде следующей схемы (рис. 1.11).

Рис. 1.11. Блок-схема алгоритма декодирования Витерби

Таким образом, АВ позволяет на длине 4…5 К тактов вычислить расстояние между принятой кодовой последовательностью и возможной кодовой последовательностью и на основе его выдать информационные символы, которые формируют данный кодовый путь.

1.5.2. Алгоритм MAP

Алгоритм MAP известен также как модифицированный алгоритм Баля под названием BCJR (Bahl-Cocke-Jelinek-Raviv) и был предложен Берроу [1, 3, 7, 10, 15, 16]. Алгоритм MAP использует «мягкую» оценку достоверности декодированного символа, а также осуществляет анализ *априорной и апостериорной вероятностей* приема верного кодового слова. *Априорной* называется информация, которой обладает декодер до приема кодового слова, а *апостериорной* – информация, которую получает декодер после обработки кодового слова. *Апостериорная информация*, полученная о кодовом слове на выходе первой итерации декодирования, поступает на вход декодера следующей итерации и является для него уже *априорной информацией*.

Алгоритм MAP, в отличие от AB, использует как минимум два пути декодирования по кодовой решетке, названные «прямым» и «обратным» путями. На рис. 1.12 отображены различные пути для «прямого» и «обратного» обхода кодовой решетки.

На рис. 1.13 представлены вероятности ошибочного перехода от одного узла кодовой сетки к другому (другим) при прямом и обратном обходе кодовой решетки. Вероятности ошибочного перехода $\alpha(m)$ при прямом обходе рассчитываются по формуле [16, 17]

$$\alpha_k(m) = \frac{\sum_{m'} \sum_{i=0}^{'} \gamma_i\left(\left(y_k^s, y_k^p\right),\ m',\ m\right)\alpha_{k-1}(m')}{\sum_n \sum_{m'} \sum_{i=0}^{'} \gamma_i\left(\left(y_k^s,\ y_k^p\right) m',\ m\right)\alpha_{k-1}(m')}, \qquad (1.2)$$

т. е. в каждом узле m вероятность представляет собой мультипозицию предыдущих вероятностей.

Способом, аналогичным $\alpha_k(m)$, рассчитываются вероятности ошибочного перехода при обратном обходе кодовой решетки:

$$\beta_k(m) = \frac{\sum_{m'} \sum_{i=0}^{'} \gamma_i(\left(y_{k+1}^s, y_{k+1)}^p\right), m',\ m)\beta_{k+1}(m')}{\sum_m \sum_{m'} \sum_{i=0}^{'} \gamma_i((y_{k+1}^s, y_{k+1}^p), m',\ m)\beta_k(m')}, \qquad (1.3)$$

где

$$\gamma_i\left(\left(y_k^s y_k^p\right) m' m\right) = p\left(\left(y_k^s y_k^p\right)\right) d_k = i \cdot m' m q(d_k = i) m' m \pi(m \mid m'). \qquad (1.4)$$

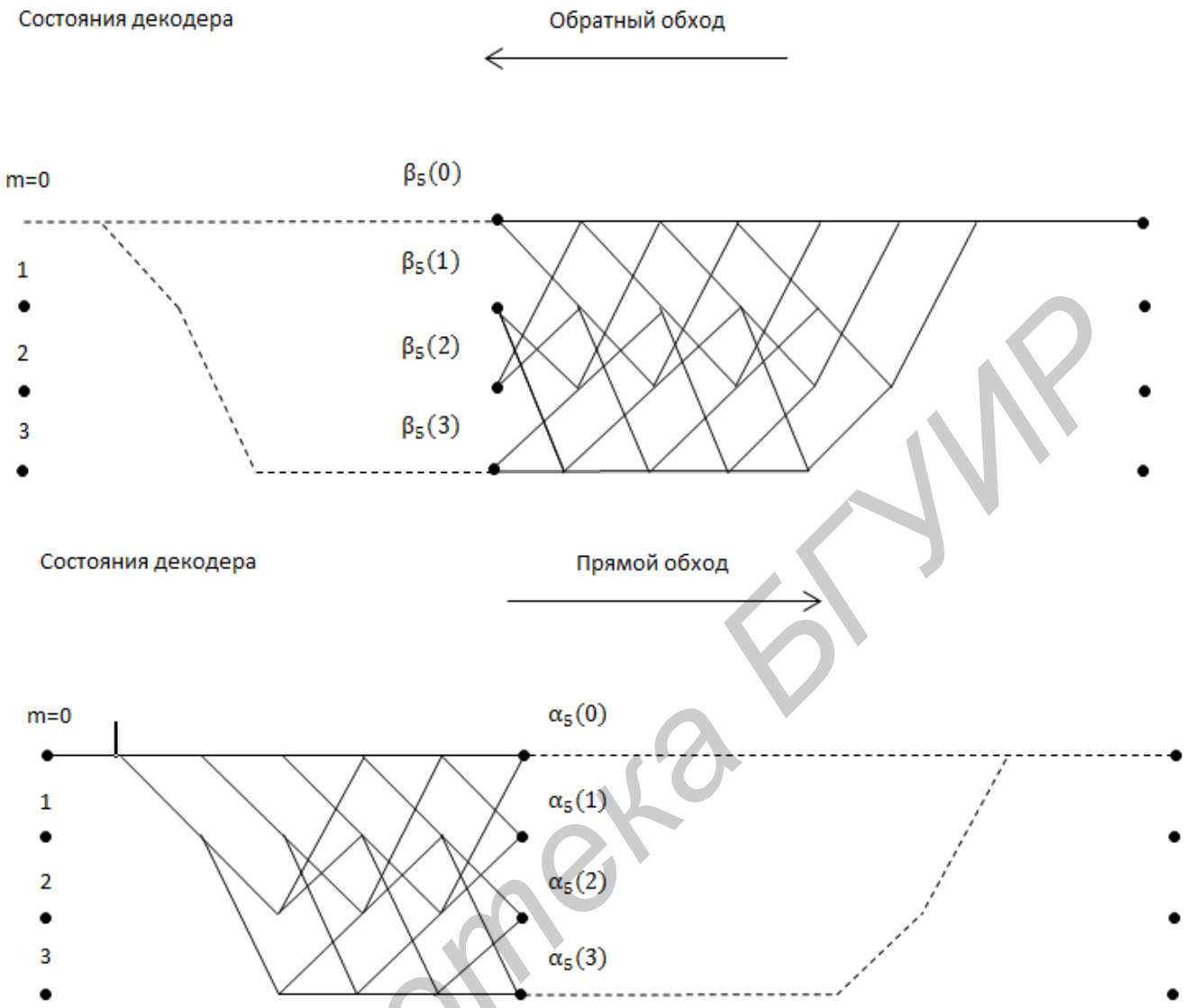


Рис. 1.12. Модифицированный алгоритм Баля

Для каждой пары $\gamma_{k-1}(m'm)$ и $\gamma_{k+1}(m',m)$ результатом становится значение $R_k = (y_k^s, y_k^p)$ и $R_{k+1} = (y_{k+1}^s, y_{k+1}^p)$ соответственно.

В выражении (1.4) вероятность изменения состояния канала связи означает в общем случае появление (y_k^s, y_k^p) при передаче (x_k^s, x_k^p). Различаются они между собой по плотности вероятности. После всех преобразований на прямом и обратном обходе все схожие вероятности объединяются в одну группу. Схожесть вероятностей определяется их численным значением и нужна только для «мягкого» отбора. Фактически сам процесс «мягкого» отбора сводится к нахождению всех логарифмических отношений правдоподобия $\Lambda(\alpha_k)$(LLR). На рис. 1.14 приведены все необходимые расчеты и направления для символов.

Отсчет $k-1, k, k+1$

Полученный код R_{k-1}, R_k, R_{k+1}

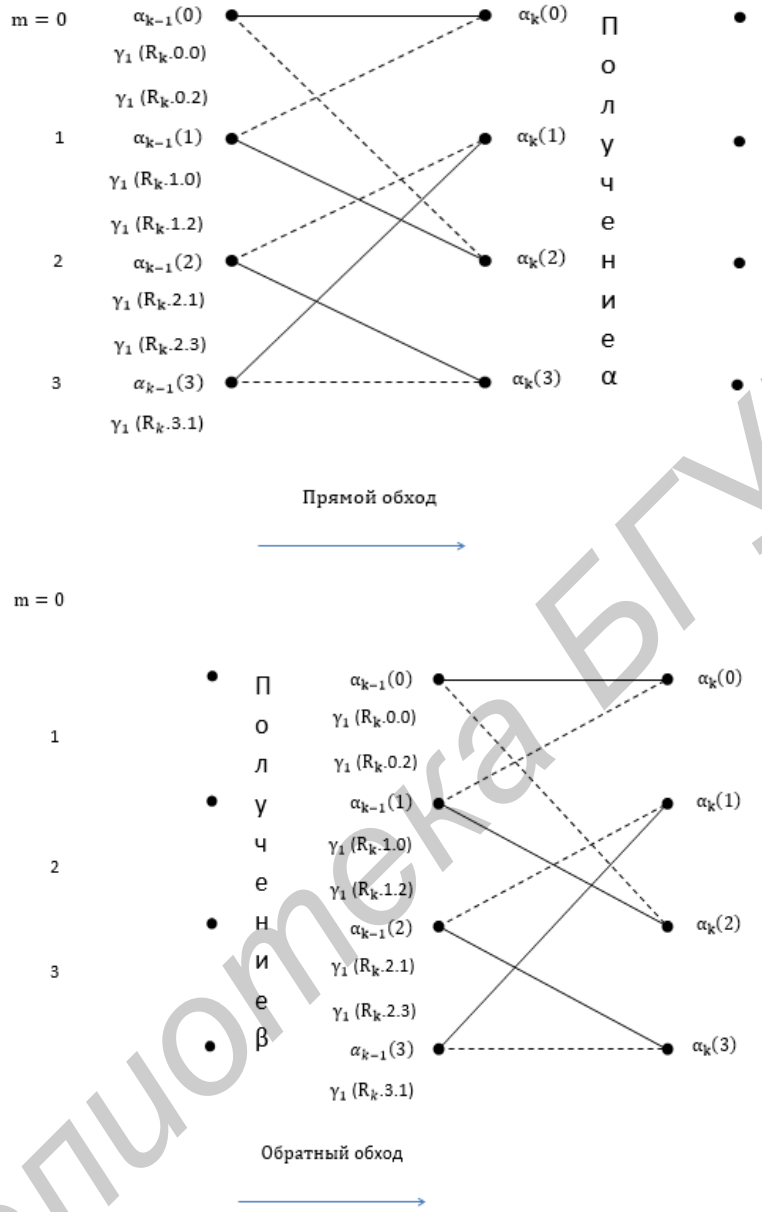
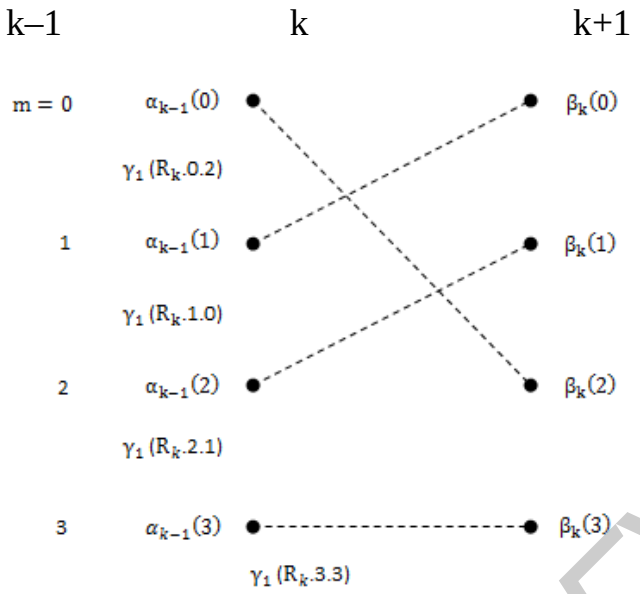


Рис. 1.13. Расчет вероятностных характеристик MAP

Суммарная вероятность в единичном бите



Суммарная вероятность в нулевом бите

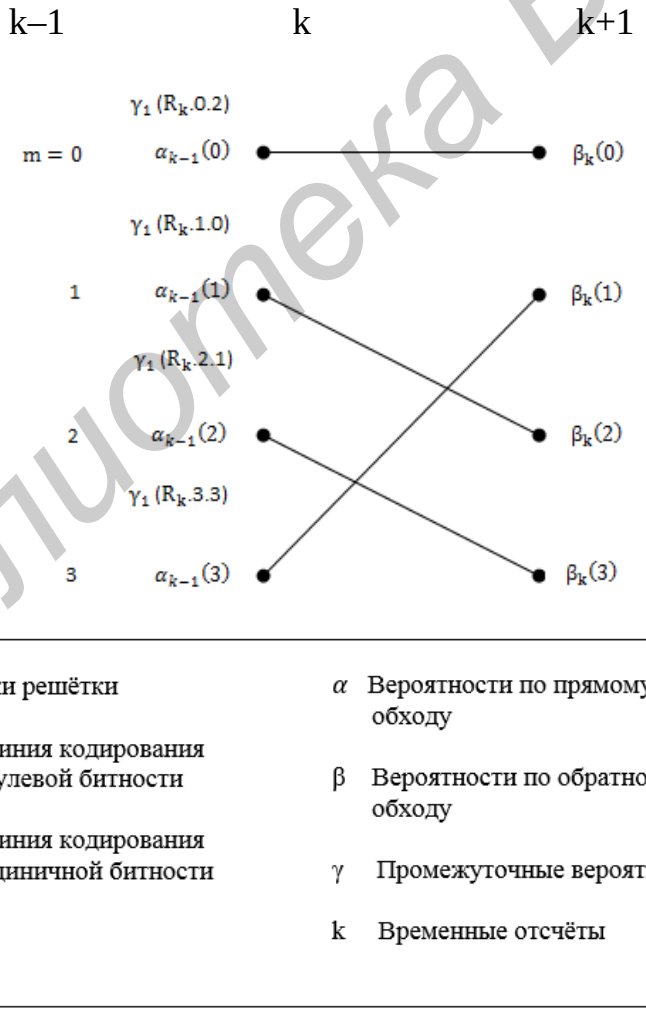


Рис. 1.14. Расчет количества подобных (равных) вероятностей

Для динамического диапазона расчет $\Lambda(d_k)$ выполняют по формуле

$$\Lambda(d_k)=\ln\frac{\sum_m\sum_{m'}\gamma_1\left((y_k^s,\ y_k^p),m',\ m'\right)\alpha_{k-1}(m')\beta_k(m)}{\sum_m\sum_{m'}\gamma_0((y_k^s,\ y_k^p),\ m',\ m)\,\alpha_{k-1}(m')\beta_k(m)}\,. \qquad (1.5)$$

Таким образом, описанный здесь MAP-алгоритм оптимален лишь для действительно большого числа подобных вероятностей. В общем случае MAP-алгоритм не принимается на практике из-за своей избыточной сложности.

Оптимальным для его применения считаются вероятности α, β, γ, лежащие в пределах 0…1, $(10^{-\infty} \ldots 10^{0})$. Однако для работы в таком большом диапазоне вероятностей ПЗУ декодеров также должны быть большими, что на практике не всегда оправдано. В связи с этим на практике зачастую от численных значений α, β, γ переходят к их логарифмам.

1.5.3. Алгоритм Max-Log-MAP

Чтобы избежать сложных математических расчетов MAP-декодирования, практически все расчеты могут быть выполнены в логарифмической области [3, 7, 8]. Кроме того, логарифм и экспоненциальные вычисления могут быть устранены путем следующей аппроксимации:

$$\max^*(x,y) \triangleq \ln(e^x + e^y)\max(x,y) + \log\left(1 + e^{-|y-x|}\right). \qquad (1.6)$$

Значения x, y можно легко рассчитать с помощью таблицы look-up-table (LUT). Логарифмические вычисления величин $L(d_k)$, $\alpha_k(S_k), \beta_k(S_k)$ и $\gamma_i(S_{k-1}, S_k)$ могут быть выполнены на основе следующих формул:

$$L(d_x) = m^*_{(S_{K-1},S_K)}\left(\overline{\gamma_1}(S_{K-1},S_K) + \overline{\alpha_{K-1}}(S_{K-1}) + \overline{\beta_K}(S_K)\right) - \max^*_{(S_{K-1},S_K,0)}(\overline{\gamma_0}(S_{K-1},S_K) + \overline{\beta_K}(S_K)); \qquad (1.7)$$

$$\overline{\alpha}_k(S_k) = \max^*(\overline{\alpha}_{k-1}(S_{k-1}) + \overline{\gamma}_i(S_k, S_{k-1})) \qquad (1.8)$$

$$\overline{\beta_k}(S_k) = \max^*{}_{(S_{k,\ i})}(\overline{\beta_{k+1}}(S_{K-1}) + \overline{\gamma}_i(S_K, S_{K-1})); \qquad (1.9)$$

$$\overline{\gamma}_i\left(S_{K-1}, S_K\right) = \frac{2}{N_0}\left(y_k^s x^s(i) + y_k^p x_k^p(i, S_{k-1})\right) + \ln\left(P(S_k|S_{K-1})\right) + K. \qquad (1.10)$$

где k – постоянная величина;

N_0 – спектральная мощность шума (при расчетах N_0 принимается постоянной, а значение k можно исключить).

К недостатку Max-Log-MAP следует отнести снижение достоверности оценки правильности принятых информационных символов. Вероятность ошибочного декодирования Max-Log-MAP увеличивается с увеличением мощности шумов.

1.5.4. Алгоритм Log-MAP

Log-MAP является упрощением алгоритма Max-Log-MAP. Упрощение было выполнено Робертсоном [15–17] на основе использования логарифма Якобсона. Это позволило избавиться от лишней аппроксимации и учитывать максимальное количество экспонент. В соответствии с [15–17] логарифм вычислений имеет вид

$$\ln(\exp(\delta_1) + \exp(\delta_2) = \max(\delta_1, \delta_2) + \ln(1 + \exp(-|\delta_2 - \delta_1|)) =$$
$$= \max(\delta_1\,, \delta_2) + f_c\,(\,|\delta_1 - \delta_2|\,), \qquad (1.11)$$

где f_c – корректирующая функция.

В процессе преобразования MAP-алгоритмов все максимизации контролировались корректирующей функцией. Однако введение этой функции также усложняет алгоритм работы декодера, за тем лишь исключением, что при верном подборе $|\delta_1 - \delta_2|$ увеличение сложности алгоритма будет минимальным. Более подробно с Log-MAP можно познакомиться в [16].

1.6. Количественные оценки турбокодов и алгоритмов декодирования

В качестве количественных параметров оценки эффективности ТК используются: вероятность ошибочного декодирования, сложность реализации, количество итераций декодирования, интервал перемежения и длина блока перфорированных информационных символов.

На рис. 1.15 приведены кривые вероятностей ошибочного декодирования ТК с параметрами: длина блока k = 65532 информационных символов; интервал перемежения $L = k + \vartheta - 1 = 65536$ ($\vartheta = 5 + 1 = 6 - \text{RSC с } R = 1/2$), k = 5 – число нулевых символов, вводимых для обнуления регистров сдвига RSC-кодеров; a = 1–18 – число итераций; алгоритм декодирования MAP; число уравнений квантования ФМ-2 – восемь [12–14].

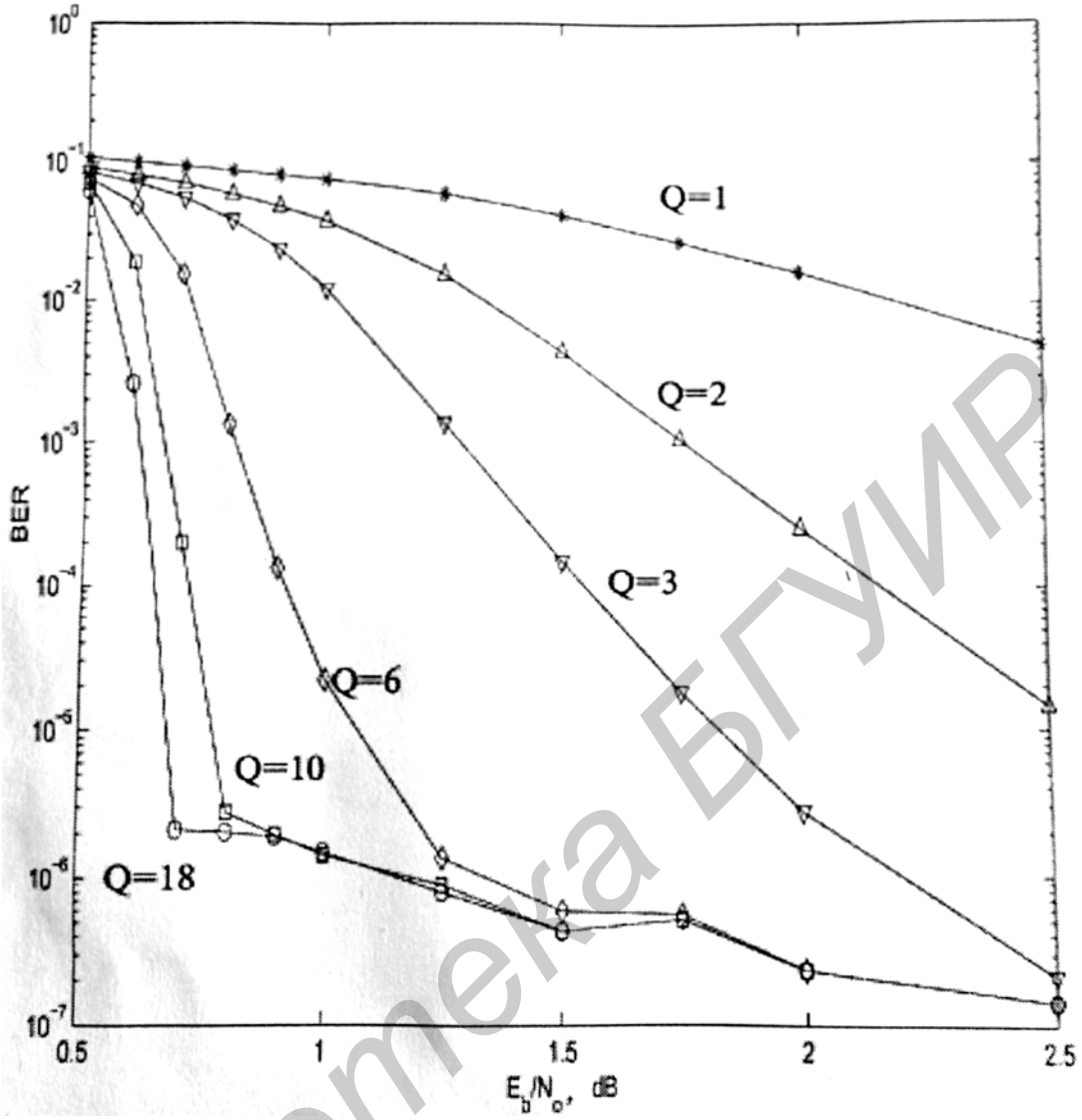


Рис. 1.15. Вероятности ошибочного декодирования ТК в зависимости от числа итераций при L = const

Из приведенных кривых вероятностей ошибочного декодирования следует, что оптимальное число итераций не превышает 10. На рис. 1.16 приведены кривые вероятности ошибочного декодирования ТК в зависимости от длины интервала перемежения информационных символов.

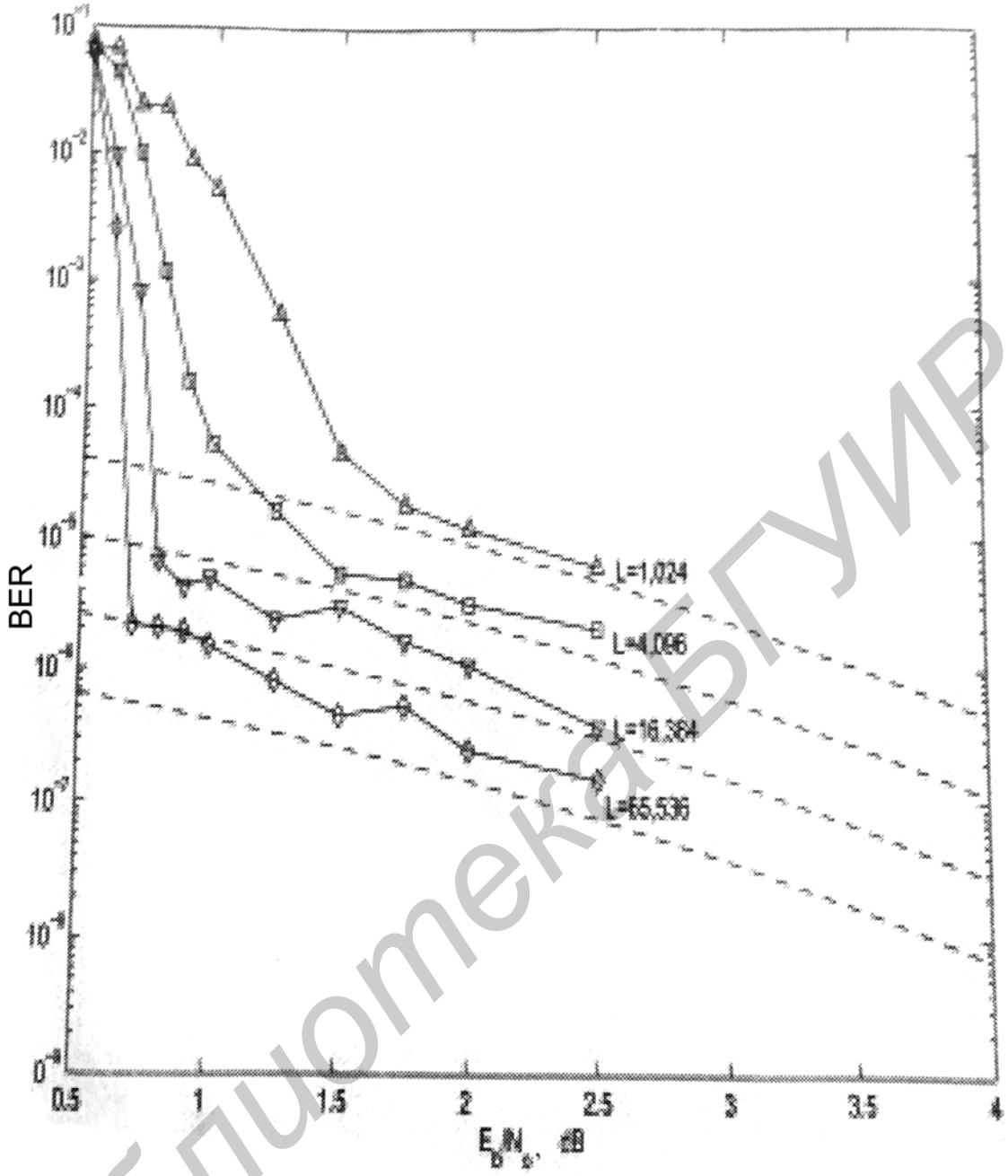


Рис. 1.16. Вероятности ошибочного декодирования ТК в зависимости от длины интервала перемежения при Q = const и методе модуляции ФМ-2

Оценка *энергетической эффективности* (ЭЭ) каналов передачи данных выполнялась при равной вероятности ошибочного декодирования ($P_{ош.декод} = 10^{-5}$).

Из результатов сравнения следует, что максимальную ЭЭ канала передачи данных обеспечивает применение ТК. *Энергетический выигрыш кодирования* (ЭВК) канала передачи данных от применения ТК составляет

$$ЭВК = (\frac{E_в}{N_0})_{БКФМ-2} - (\frac{E_в}{N_0})_{К.ФМ-2} = 9{,}6 - 0{,}7 = 8{,}9 \text{ дБ}, \quad (1.12)$$

где $(\frac{E_в}{N_0})_{БКФМ-2} = 9{,}6$ – требуемое отношение для обеспечения $P_{ош.дек} = 10^{-5}$ при использовании некодированной ФМ-2;

$(\frac{E_в}{N_0})_{К.ФМ-2}$ – кодированная ФМ-2 .

Таблица 1.1

Сравнительная оценка энергетической эффективности турбокодов и сверточных кодов

Вид СКК	E_b/N_0 , дБ
Турбокод (n = 131072, k = 65532) с BPSK	0,7
Сверточный код с k = 32 и с последовательным декодированием (использовался для связи с космическими аппаратами Pioneer-10, 11 и 12 для полетов на Юпитер, Сатурн и Венеру соответственно) и BPSK	2,7
Сверточный код с k = 9 (используется в Globarstar*), декодированием по Витерби и BPSK	3,5
Сверточный код Оденвальдера с k = 7 и с алгоритмом декодирования Витерби (DVB-S, Intelsat) и BPSK	4,5
Сверточный код с k = 5, алгоритм декодирования Витерби (используется в GSM*) и BPSK	5,3
Некодированная BPSK	9,6

На рис. 1.17 приведены кривые вероятностей ошибочного декодирования ТК при реализации алгоритмов декодирования MAP, Log-MAP и Max-Log-MAP при использовании RSC с R = 1/2 с G = {23, 37}, 8 итераций, длине интервала перемежения N = 1024 и типе ДКС с АБГШ.

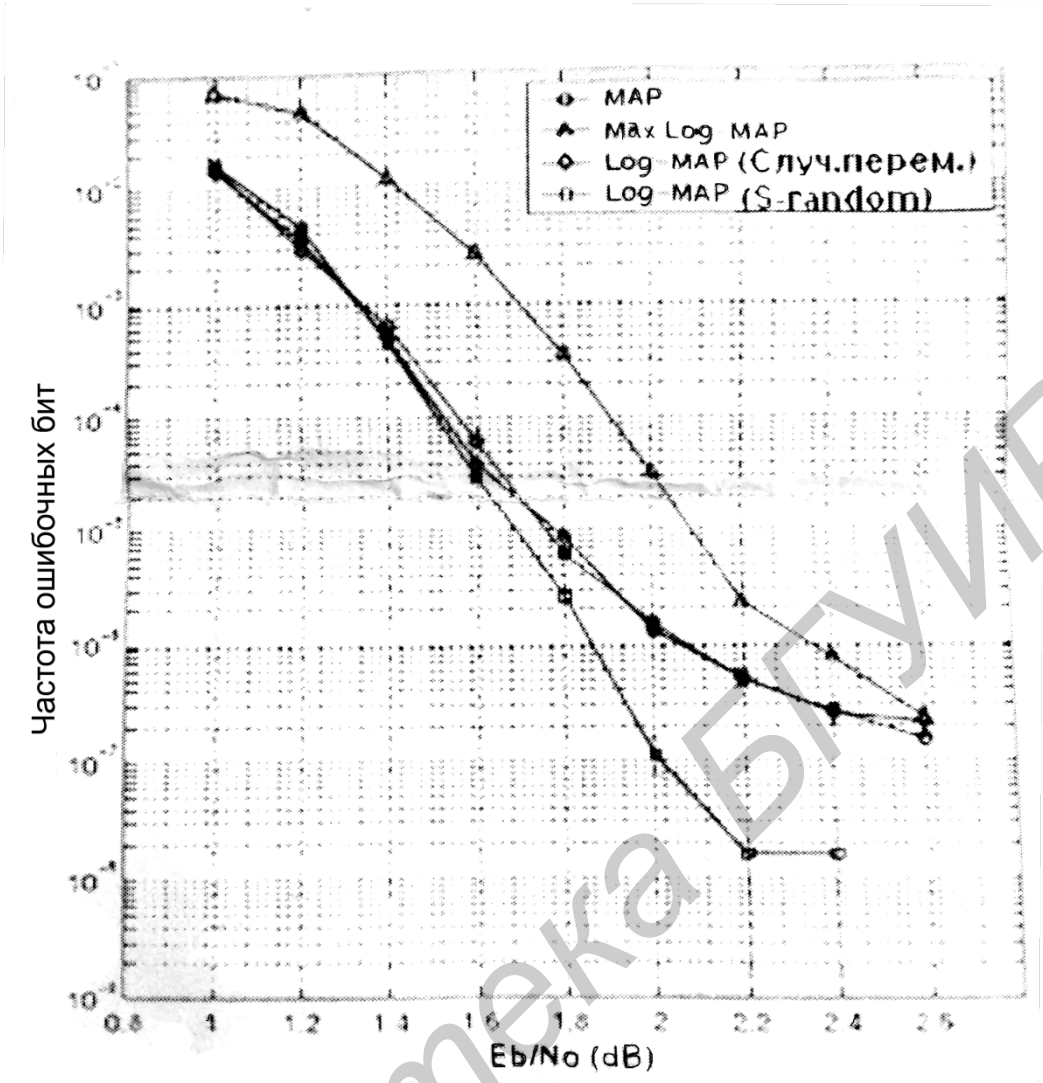


Рис. 1.17. Кривые вероятности ошибочного декодирования ТК при реализации алгоритмов декодирования MAP, log-MAP и MAP-Log-MAP

Из рис. 1.17 следует, что алгоритмы MAP и LogMAP обеспечивают практически одинаковую вероятность ошибочного декодирования. Вероятность ошибочного декодирования алгоритма Log-MAP приведена при использовании перемежителя S-Random и S – случайного перемежителя. Более простой алгоритм декодирования MAP-Log-MAP проигрывает S-случайному перемежителю порядка 0,4 дБ при низких отношениях $E_в/N_0$.

Чтобы иметь полное представление об эффективности ТК, необходимо определить сложность его реализации, а именно сложность реализации канального декодера.

Сравнение сложности реализации турбокодов чаще всего выполняется по сравнению со сложностью реализации декодера СК, реализующего алгоритм Витерби, и реже со сложностью реализации каскадных кодов, включающих

внешние коды Рида – Соломона и сверточные внутренние коды. Каскадные коды в настоящее время являются важнейшей составной частью спутниковых и космических систем связи для передачи информации Земля – Космос.

Следует отметить, что определение сложности реализации алгоритмизации алгоритма декодирования – неточная наука. Это определяется следующими причинами.

Во-первых, существуют различные определения «сложности». Определение может зависеть от метода реализации. Например, в случае аппаратной реализации оборудование (обычно) уже определено и задача программиста – сократить сложность алгоритма до наименьшего числа вычислений, чтобы алгоритм в устройстве обработки сигналов работал как можно быстрее. Программист будет стараться оптимизировать код, используя инструкции, которые, как известно, принимают меньше инструкционных циклов. В случае же программной реализации разработчик может спроектировать устройство, устроенное так, чтобы свести к минимуму общую задержку, связанную с алгоритмом, так что все будет выполняться как можно быстрее, или он/она может стремиться к уменьшению общего количества ячеек, нужного для реализации алгоритма, чтобы достичь компактной конструкции.

Для оценки сложности реализации декодеров ТК и Витерби учитывается необходимое количество операций *сложения/вычитания/умножения/деления/сравнения* для декодирования или определения полярности символа или бита сообщения в аппаратной реализации декодеров.

Количество операций для обоих декодеров зависит от числа состояний M в решетках декодера. В случае декодера Витерби оно также зависит от *глубины пути декодирования* D: D – это размер окна (интервала), использующегося для хранения выживших путей в решетках. В алгоритме Витерби один выживший путь определен в каждом состоянии решетки, затем этот путь должен быть обновлен в каждом предшествующем состоянии через расстояние D в решетках. Расстояние D обычно выбирается примерно в 5 раз больше, чем конструктивная длина K кода, чтобы остающиеся пути сводились в одну точку. Это дает возможность точно определить выходной битовый поток. Для обоих декодеров число состояний решеток M относится к конструктивной длине K как $M = 2^{(K-1)}$. Итак, $K = \log_2 M + 1$ и $D = 5 \times K = 5 \log_2 M + 5$. Число вычислений разбивается дальше на количество операций СЛОЖИТЬ/ВЫЧИСЛИТЬ, УМНОЖИТЬ/РАЗДЕЛИТЬ и СРАВНИТЬ, операции СРАВНИТЬ включают в себя такие условные операции ветвления, как IF/ELSE, FOR и WHILE.

Для операции DSP (аппаратной реализации) под сложностью имеется в виду число инструкционных циклов, необходимых для выполнения операций. DSP разработана для очень быстрого выполнения большого числа сложений и умножений: устройства обработки сигналов обычно выполняют отдельную операцию сложения за один инструкционный цикл.

На рис. 1.18 приведены кривые зависимости сравнения сложности в вычислениях от символа (бит) турбодекодера (два RSC-декодера с алгоритмом Log-MAP, Q = 8) и декодера Витерби (ϑ = 5 и D = 25) [12–14].

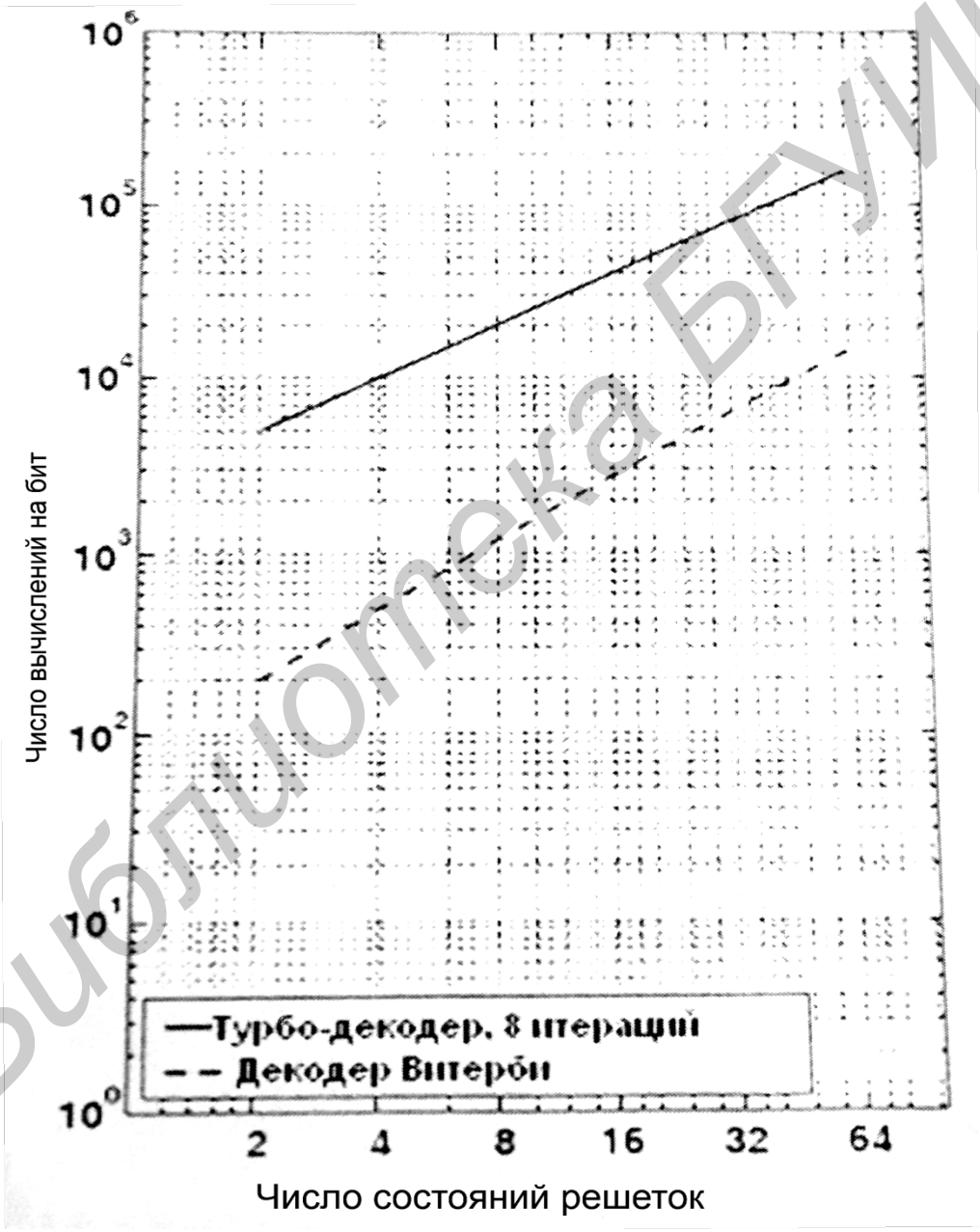


Рис. 1.18. Кривые зависимости сложности сравнения вычислений от символа (бит) турбодекодера и декодера Витерби

В табл. 1.2 приведено общее число вычислений, необходимых для декодирования одного символа при использовании турбокода с M = 16 и СК с D = 25.

Таблица 1.2

Сложность вычислений в расчете декодирования одного символа (бита)

ДЕКОДЕР	СРАВНИТЬ	УМНОЖИТЬ/ РАЗДЕЛИТЬ	СЛОЖИТЬ/ ВЫЧЕСТЬ	ЗАГРУЗИТЬ/ СОХРАНИТЬ	ИТОГ
Log-MAP – 8 итераций – 2 декодера	76M = = 76·16 = = 1216	6M = 6·16 = = 96	52M = 52·16 = = 832	22M = 22·16 = = 352	134M = 2144 124M = 19968 2496M = 39936
Витерби D = $= 5\log_2 M + +5$	M(10 + + 2D) = = 960	11M = 11·16 = = 176	M(11+D) = 576	M(17 + 2D) = = 1139	M(49 + 5D) = = 2784 M(74+ $+ 25\log_2 M$)

Из приведенных результатов оценки сложности реализации анализируемых турбокодов и декодера Витерби следует, что число вычислений, осуществляемых в расчете на декодирование одного кодового (информационного) символа (бита) выше, поэтому реализация данного турбокодера сложнее в 39936/2784 = 14,3 раза, чем декодера Витерби. Однако турбодекодер с M = 4 состояниями (два декодера с Log-MAP и Q = 8 итерациями) имеет практически равную сложность реализации с декодером Витерби с 64 состояниями.

В целом следует отметить, что эффективность турбодекодеров увеличивается с увеличением числа итераций и длины интервала перемежения. Максимальный прирост достоверности передачи информации или уменьшение вероятности ошибочного приема информации обеспечивается на начальных этапах итерационного процесса. Турбокоды эффективно могут быть использованы в системах передачи, обработки и хранения информации, при требующих минимальной вероятности ошибочного приема ($P_{ошиб.доп} \leq \leq 10^{-12}$) данных и допускающих большую задержку, вносимую при декодировании.

2. ПЕРЕМЕЖИТЕЛИ ИНФОРМАЦИОННЫХ СИМВОЛОВ КАНАЛЬНЫХ КОДЕКОВ ТУРБОКОДОВ

2.1. Общая классификация и характеристика перемежителей

Известно [14, 17, 19], что использование помехоустойчивого кодирования с коррекцией случайных (независимых) ошибок в каналах связи с группированием ошибок является нецелесообразным ввиду того, что не будет осуществляться полная корреляция ошибок. Применение кодов с высокой корректирующей способностью имеет два недостатка [1–5]:

– коды имеют высокую избыточность;

– высокая сложность реализации и большой объем оборудования.

Применение кодов, рассчитанных для коррекции случайных или независимых ошибок, в каналах связи с группированием ошибок (в каналах с памятью) возможно при совместном использовании с устройствами перемежения и деперемежения (деперемежения).

Обобщенная структурная схема канала передачи данных с использованием деперемежителей/перемежителей приведена на рис. 2.1.

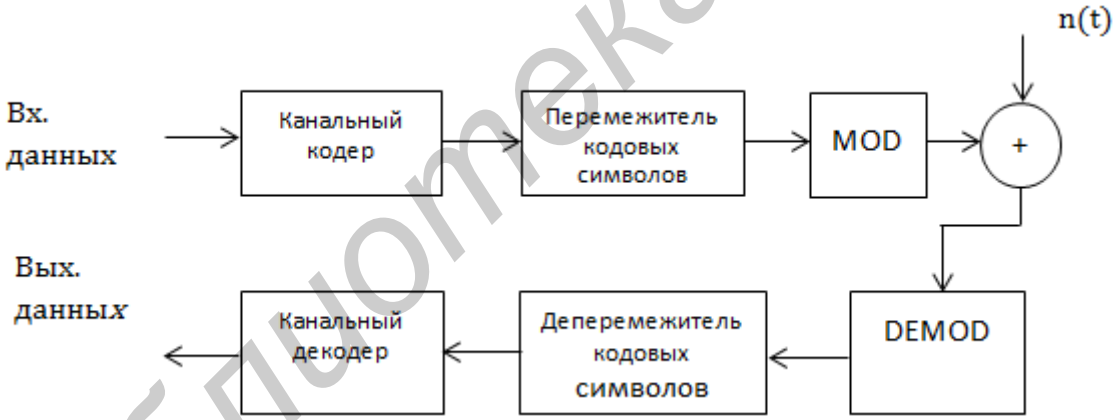


Рис. 2.1. Обобщенная структурная схема канала передачи данных с использованием деперемежителей/перемежителей

Назначение деперемежителей/перемежителей в данном канале связи состоит в том, чтобы преобразовать канал связи с группированием ошибок в канал связи со случайными ошибками, гарантированно корректируемыми используемым помехоустойчивым кодом. Особенность включения деперемежителя/перемежителя в данной структурной схеме канала связи состоит в том, что осуществляется перемежение кодовых символов.

Выбор конструкции и параметров деперемежителя/перемежителя зависит как от модели дискретного канала связи (ДКС), так и от типа помехоустойчивого кода и реже от типа модуляции.

Все известные деперемежители/перемежители делятся на две группы [16, 20–23]:

– *периодические;*

– *псевдослучайные*.

Периодическими деперемежителями/перемежителями называются такие устройства, у которых перестановка информационных или кодовых символов соответствует периодической функции времени.

Данные деперемежители/перемежители являются более простыми как в работе, так и в реализации, но обладают меньшими свойствами адаптации к характеристикам канала связи.

Псевдослучайные деперемежители/перемежители перестановку информационных (кодовых) символов осуществляют по псевдослучайному закону. Данные деперемежители/перемежители обладают обратными качествами по отношению к периодическим.

Периодические деперемежители/перемежители по конструкции делятся в основном на два типа [16, 19, 24]:

– *блочные (блоковые);*

– *треугольные*.

Различие между данными деперемежителями/перемежителями очень похоже на различие между блоковыми (групповыми) и сверточными (непрерывными) помехоустойчивыми кодами.

2.1.1. Блочный (блоковый) перемежитель

Блочный (блоковый) перемежитель выполняется в виде матрицы (таблицы), содержащий n_1 строк и n_2 столбцов, при этом возможны варианты построения перемежителей с $n_1 = n_2$ и $n_1 > n_2$. На рис. 2.2 приведена обобщенная структурная схема блочного перемежителя с $n_1 = n_2$.

При использовании блочных перемежителей передаваемые информационные (кодовые) символы записываются первоначально по строкам или столбцам, а считывание символов производится в обратном порядке. Современные блочные деперемежители/перемежители выполняются в виде оперативных запоминающих устройств (ОЗУ), обеспечивающих одновременно запись и считывание передаваемых информационных (кодовых) символов.

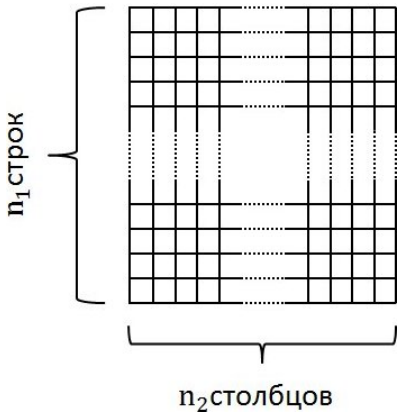


Рис. 2.2. Обобщенная структурная схема блочного перемежителя с $n_1 = n_2$

В матрице блочного (блокового) перемежителя должно содержаться достаточное количество столбцов, чтобы могли поместиться все возможные ошибочные символы. Количество строк определяется необходимостью определения расстояния между ошибочными символами, которые декодер может исправить. На рис. 2.3, а приведен пример построения бокового перемежителя с размером матрицы 6×4.

а

1	5	9	13	17	21
2	6	10	14	18	22
3	7	11	15	19	23
4	8	12	16	20	24

б

1	5	9	13	17	21
2	6	10	**14**	**18**	**22**
3	**7**	11	15	19	23
4	8	12	16	20	24

в

1	5	9	13	17	21
2	6	10	**14**	**18**	**22**
3	**7**	**11**	**15**	**19**	**23**
4	8	12	16	20	24

Рис. 2.3. Обобщенная структурная схема блокового перемежителя:
а – исходная матрица; б – матрица с пятью ошибками;
в – матрица с девятью ошибками

Пять ошибочных информационных символов, которые условно может исправить декодер, подчеркнуты и выделены полужирным (рис. 2.3, б). На рис. 2.3, в показаны девять ошибок, с которыми декодер не может справиться ввиду того, что в матрице перемежителя 6×4 такое количество ошибок не может поместиться ни в строку, ни в столбец. В этом случае отделить ошибочные символы от правильных никаким перемежением не получится, в отличие от случая на рис. 2.3, в, где их можно расположить с минимальным кодовым расстоянием, равным двум.

Термин «идеальное перемежение» относится к такому процессу перемежения, который приводит к затуханию некоррелированных связей между соседними символами кодовой последовательности. Однако реализовать такую модель перемежения информационных (кодовых) символов практически невозможно. Разработка перемежителя, максимально приближенного к идеальному перемежителю, является актуальной задачей теории и практики как помехоустойчивого кодирования, так и электросвязи.

Основными свойствами блочных деперемежителей являются:

а) любой пакет ошибок длиной (кратностью) $t_n = n_2$ при $n_1 = n_2$ и записи символов по строкам после перемежения распределяется в одиночные ошибки на длине n_1 информационных (кодовых) символов: если все $t_n = n_2$ символов являются ошибочными, то такой пакет ошибок считается «плотным»;

б) любой пакет ошибок кратностью $t_n = \delta \cdot n_2$, $\delta \geq 2$ после деперемежения распределяется в пакетные ошибки кратностью δ символов, разделенных безошибочным интервалом в $n_1 - \delta$ символов;

в) общий интервал деперемежения символов, на который распределяется пакет ошибок кратностью $t_n = n_2$, составляет $И = n_1 \cdot n_2$ символов при $n_1 \neq n_2$ и $И = n_1^2$ символов при $n_1 = n_2$;

г) периодическая последовательность одиночных ошибок, разделенных n_1 безошибочными символами, после деперемежения переходит в пакет ошибок кратностью $t_n = n_2$ при $n_1 = n_2$;

д) задержка информации после деперемежения составляет $L_{зад} = 2 \cdot n_1 \cdot n_2$ символов, а для реализации перемежителя и деперемежителя требуется по $n_1 \cdot n_2$ ячеек памяти;

е) в практических системах связи значения n_1 и n_2 выбираются такими, чтобы пакет ошибок кратностью t_n символов был $\leq n_2$;

ж) если же характеристики канала связи нестационарны, то деперемежитель/перемежитель, согласно свойству «б», будет работать неустойчиво.

Важнейшими недостатками блочных деперемежителей/перемежителей являются большая задержка информации при декодировании и высокая сложность организации цикловой синхронизации перемежителя и деперемежителя. Перемежитель и деперемежитель могут быть синхронизированы путем применения стандартной кадровой (цикловой) синхронизации, когда специальное синхрослово с «хорошими» корреляционными свойствами периодически «вставляется» в кодовую последовательность в устройстве перемежения, а затем восстанавливается (исключается) устройством цикловой синхронизации (УЦС) в устройстве деперемежения. При таком методе организации цикловой синхронизации (ЦС) деперемежителя/перемежителя в кодовую последовательность дополнительно «вставляется» (включается) от 1 до 2 процентов избыточной синхроинформации.

Другой способ организации ЦС деперемежителей/перемежителей состоит в том, что некоторые кодовые символы заменяются синхросимволами, которые затем «стираются» (переключаются) на входе декодера и заменяются нулевыми символами. Данный способ позволяет избежать увеличение избыточной информации в передаваемых кодовых словах, но увеличивает вероятность ошибочного декодирования. Для уменьшения вероятности ошибочного декодирования необходимо, чтобы стертые символы находились далеко друг от друга на входе декодера. При уменьшении $P_с/P_ш$ на входе демодулятора ДКС сложность построения УЦС деперемежителей/перемежителей увеличивается.

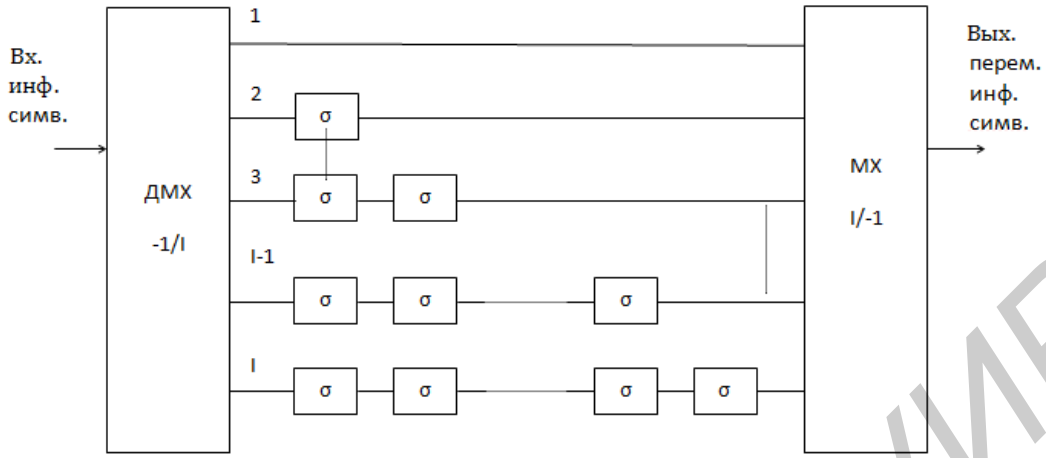
Блочные (блоковые) деперемежители/перемежители целесообразно использовать с блочными (блоковыми) помехоустойчивыми кодами, а со сверточными (непрерывными) помехоустойчивыми кодами наиболее оптимально согласуются по параметрам деперемежители/перемежители треугольной конструкции или конструкции Рамсея [16, 23, 25].

2.1.2. Треугольный перемежитель

Треугольный перемежитель выполняется в виде I ($I \gg 2$) параллельных цепей передачи информационных (кодовых) символов, имеющих разную задержку информационных (кодовых) символов.

На рис. 2.4 представлены соответственно структурные схемы перемежителя и деперемежителя конструкции Рамсея [1, 6].

а



б

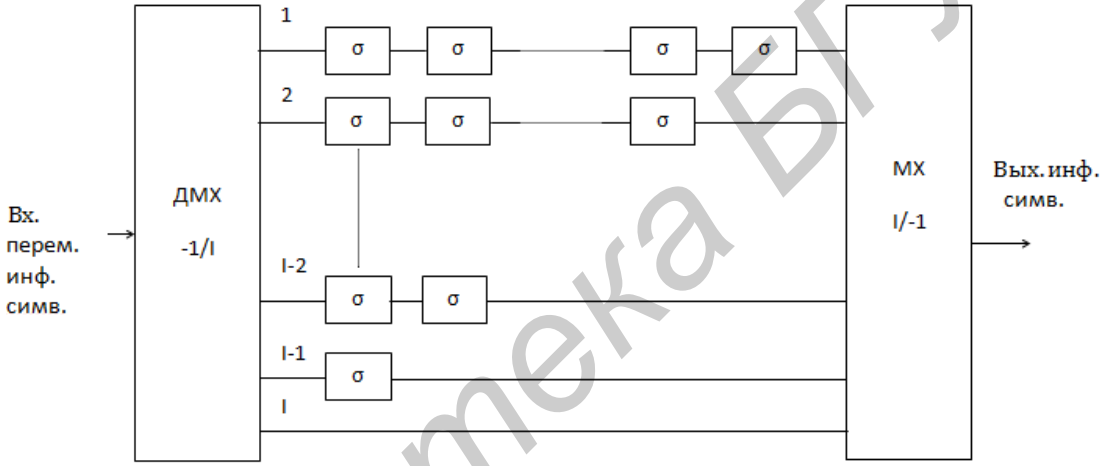


Рис. 2.4. Обобщенная структурная схема деперемежителя/перемежителя треугольной конструкции:
а – перемежитель; б – деперемежитель

Интервал перемежения информационных (кодовых) символов определяется двумя параметрами, а именно: I – количеством параллельных потоков, на которое распределяется поток входных информационных (кодовых) символов, и величиной начальной задержки символов во втором подпотоке. Символы первого подпотока передаются без задержки. Максимальная задержка символов обеспечивается последним подпотоком и составляет для данной схемы перемежителя $(I - 1)\cdot\delta$ тактов. Деперемежитель имеет обратный принцип построения перемежителя: в первом подпотоке обеспечивается максимальная задержка символов ($(I - 1)\cdot\delta$ тактов), а в последнем подпотоке символы передаются без задержки.

Интервал перемежения символов равен $h = \delta \cdot I$ тактов. Задержки символов в последующих подпотоках составляют 2δ, 3δ, …, $(I - 1) \cdot \delta$ тактов. При $I = n_A$ – длине кодового ограничения СК и $\delta = 1$ пакет ошибок кратностью $t_n = n_A$ кодовых символов после деперемежения распределяется в одиночные ошибки на длине кодового ограничения каждой из $I = n_A$ кодовых ограничений.

Следовательно, пакет ошибок кратностью $t_n = n_A$ после деперемежения распределяется в одиночные ошибки на интервале в $И = I \cdot n_A = n_A \cdot n_A = n_A^2$ кодовых символов. Пакет ошибок кратностью $t_n = 2 \cdot n_A$, $3 \cdot n_A$, …, $k_0 \cdot n_A$, $(k_0 \geq 4)$, возникающий на интервале $И = n_A$ кодовых символов, после деперемежения распределяется в двух-, трех- и k_0-кратные ошибки на длине каждой из n_A кодовых ограничений.

Параметры деперемежителя I и δ выбираются аналогичным образом, как и для блочных де/перемежителей, т. е. исходя из модели ДКС и корректирующей способности СК, а именно:

а) число параллельных подпотоков I перемежителя должно быть равно кратности t_n и $\delta = 1$. В этом случае пакет ошибок t_n распределяется на одиночные ошибки на длине n_A и использовать СК, корректирующий одиночные ошибки, что будет обеспечивать минимальную сложность реализации кодека. Следовательно, если принять $I = t_n/2$, то для получения одиночных ошибок на длине n_A необходимо первоначальную задержку увеличить в два раза, т. е. $\delta_1 = 2$;

б) если принять $I = t_n$, $\delta_1 = 2$ такта, то пакет ошибок кратностью t_n символов распределится в одиночные ошибки на длине n_A, которые будут чередоваться, т. е. одна n_A будет принята с ошибкой, а следующая – без ошибок. В данном случае можно увеличить кратность пакета ошибок в два раза, который после деперемежения распределится в одиночные ошибки в каждой n_A и на интервале $И = 2 \cdot \delta \cdot I$ кодовых символов;

в) периодическая последовательность одиночных ошибок, находящихся на расстоянии I + 1 символов друг от друга, после деперемежения переводится в пакет ошибок кратностью $t_n = I$ символов;

г) общая первоначальная задержка перемежаемых символов перемежителем и деперемежителем составляет $L_{задер} = I(I - 1) \cdot \delta$ тактов;

д) для реализации перемежителя и деперемежителя требуется одинаковое количество ячеек памяти, а именно $Q = I(I - 1)$, при $\delta_1 = 1$ – это в два раза меньше, чем требуется для реализации блочного де/перемежителя при равных интервалах перемежения;

е) деперемежители/перемежители треугольной конструкции не требуют организации отдельных УЦС или устройств ветвевой синхронизации (УВС), так как для организации ветвевого синхронизма перемежителя и деперемежителя используется УВС декодера. Однако в этом случае необходимо учитывать, что время установления и время восстановления ветвевого синхронизма кодека СК увеличивается в I раз.

В целом следует отметить, что рассмотренные блочный и треугольный деперемежители/перемежители практически применяются в каналах связи при использовании каскадных кодов и практически не пригодны в каналах связи при использовании турбокодов. Это определяется тем, что данные перемежители не обеспечивают формирование кодовых слов с малым взаимным расстоянием или кодовых слов с большими весами. Данную задачу решают деперемежители/перемежители особых конструкций или структур.

2.2. Перемежители турбокодов

2.2.1. Термины и параметры перемежителей турбокодов

Хегор и Уикер в [20–23] ввели два параметра, а именно *дисперсию* и *s-параметр*, которые определяют перемежающие свойства перемежителей. Сущность этих параметров состоит в следующем. Пусть некоторый перемежитель имеет интервал перемежения N символов и содержит n_1 строк и n_2 столбцов. Далее вводится понятие *пары позиций символов* (бит) в перемежителе, обозначим их как (i, j), где i = 0, …, N – 1, j = 0, …, N – 1, $i \neq j$, которые будут определять позиции пар символов во входной информационной последовательности. Следовательно, можно определить пары ε(i), ε(j) как позиции этих символов в выходной (перемеженной) информационной последовательности. Разница в позициях между парой входных символов и парой выходных символов обозначается соответственно как | i – j | и |ε(i) – ε(j).

Далее определим *фактор распространения* (spreadingfactors) для пары символов. Установлено, что перемежитель имеет *фактор распространения* (s, t), если $|i - j| < s$, тогда $|\varepsilon(i) - \varepsilon(j)| \geq t$. То есть, если два символа входят (отстоят друг от друга) в перемежитель меньше, чем на s позиций друг от друга, то когда эти символы будут выходить из перемежителя, они будут отстоять друг от друга на t позиций (t символов, или бит). Это соотношение является симметричным в том смысле, что каждый раз, когда $|\pi(i) - \pi(j)| < t$, $|i - j| \geq s$.

Важным свойством, связанным с фактором перемежения, является s-параметр. Он определяется как максимальное значение s, такое, что s<t. Если перемежитель имеет *s-параметр* s_0, то он имеет *фактор распространения* ($s = s_0$, $t = s_0$). Таким образом, если два символа данных входят в перемежитель менее чем на s_0 позиций друг от друга, то они выйдут по крайней мере на s_0 позиций, отключающих друг от друга.

Можно сделать запись всех различий в положении между парами входных символов $\Delta_x = |i - j|$ и выходных символов $\Delta_y = |\pi(i) - \pi(j)|$ для перемежителя и сохранять их в виде набора *векторов смещения*. Они определяются следующим образом:

$$D = \{(\Delta x, \Delta y) \in Z^2 | \Delta x = |i - j|, \Delta y = |\varepsilon(i) - \varepsilon(j)|, 0 \leq i < j < N\}, \quad (2.1)$$

где Z^2 представляет собой совокупность всех целочисленных пар.

Например, тривиальная тождественная перестановка $\varepsilon(i) = j$ имеет самый короткий набор векторов смещения:

$$D = \{(1,1), (2,2), \dots, (N-1, N-1)\}. \quad (2.2)$$

Согласно биноминальной теореме есть $|D| = N \cdot (N - 1)/2$ комбинаций позиций для пары (i, j). Длина набора векторов смещения является полезным показателем «случайности» перемежителя. Назовем эту длину *дисперсией*перемежителя, т. е. $\Gamma \equiv |D|$. *Дисперсия* лежит в диапазоне

$$N - 1 \leq \Gamma \leq N \cdot (N - 1)/2 . \quad (2.3)$$

Для преобразования значений дисперсии в удобном диапазоне определим нормированную *дисперсию* γ как

$$\gamma = 2\Gamma/(N \cdot (N - 1)), \quad (2.4)$$

которая имеет значения в диапазоне $2/N \leq \gamma \leq 1$. Следовательно, случайные перемежители, как правило, обладают высокой дисперсией, в то время как перемежители с высокой степенью регулярности, например блоковые, имеют очень низкую *дисперсию*.

2.2.2. Классификация и краткая характеристика перемежителей турбокодов

Наиболее известными перемежителями турбокодов являются [7, 26–29]:

– блочные (блоковые) перемежители;

– перемежитель Берроу – Главье;

– случайные перемежители;

– псевдослучайные перемежители;
– UMTS-перемежители;
– перемежитель «строка за строкой»;
– перемежитель четыре на четыре;
– Уорк-перемежитель.

Блочный (блоковый) перемежитель

Простейший нетривиальный перемежитель – *блоковый перемежитель* (рис. 2.5), в котором данные передаются n_1 строкам посредством n_2 столбцов прямоугольной матрицы и считываются по столбцам. Классический слева направо/сверху вниз блоковый перемежитель может быть выражен следующим образом:

$$\pi(i) = (R \cdot i) \bmod (N - 1),\ 0 \le i < N - 1,\ \pi(N - 1) = N - 1, \qquad (2.5)$$

где N – длина перемежителя.

Блоковые перемежители конструктивно просты. Тем не менее широко признанные [23] проблемы с блоковыми перемежителями как компонентами турбокодов заключаются в том, что они, как правило, допускают высокую кратность общих конечных кодовых слов. То есть всегда есть большое количество ошибочных последовательностей, самоограниченных для обоих компонентных кодов RSC_1 и RSC_2, и это может привести к низкому весу кодовой последовательности. На рис. 2.5 приведены примеры таких информационных (кодовых) последовательностей для турбокода с порождающим номиналом G = {23;37} = {010011;011111} = {1 + D^3 + D^4; 1 + D^1 + + D^2 + D^3 + D^4}. Видно, что информационные (кодовые) последовательности читаются одинаково по рядам (вход) и по столбцам (выход) и что линейные сдвиги таких последовательностей в блоковом перемежителе будут иметь тот же эффект. Блоковый перемежитель обладает высоким s-*параметром* и очень низкой *дисперсией* (низкая степень случайности).

Каждая строка и каждый столбец рис. 2.5 формирует самоограниченные последовательности. Например, последовательность {…000110101000…} имеет вес данных 4 и паритетный вес, равный 2, для первого компонентного RSC_1. После перемежения на вход второго RSC-декодера поступит аналогичная последовательность данных {…000110101000…} с весом 4 и паритетным весом 2. Для скорости передачи турбокода R = 1/3 вес кода первой строки ошибочной кодовой последовательности будет равен 4 + 2 + 2 = 8. С четырьмя такими строками, как показано на рис. 2.5, а общий вес кодовой последовательности будет равен 8·4 = 32. Кодовые последовательности с другими весами для данного ТК приведены в табл. 2.1.

Таблица 2.1

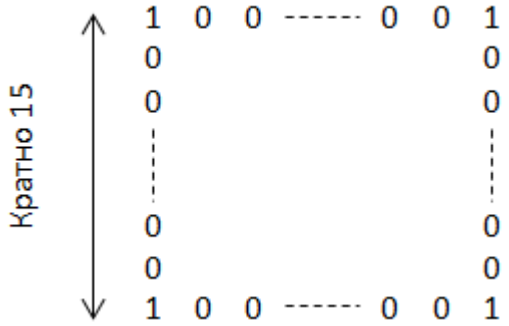
Ошибочные кодовые последовательности блокового перемежителя

Количество последовательностей	Вес кода
53 блока с весом данных 16	32
14 блоков с весом данных 12	38
6 блоков с весом данных 9	39
6 блоков с весом данных 20	50

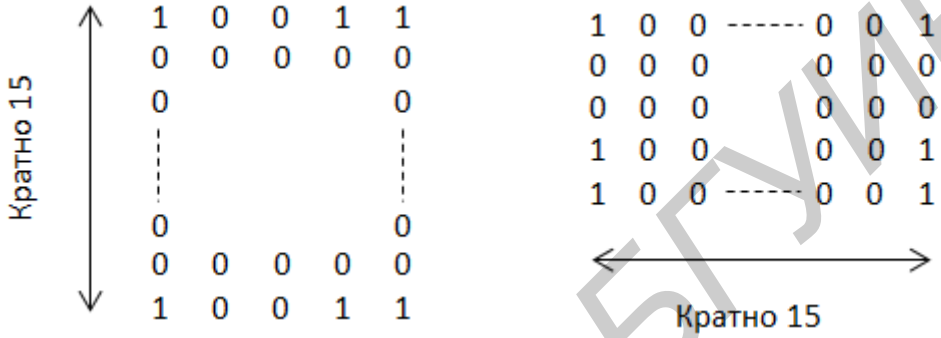
Примечание. 100 ошибочных блоков данных содержат 1486 ошибок. В среднем 14,86 ошибок на блок.

Из рис. 2.5 и табл. 2.1 следует, что перемежитель формирует переменные информационные последовательности с низким весом для данного ТК с R = 1/3. Большинство случайно выбранных символов при использовании в 1024-символьном перемежителе генерирует переменные последовательности с более высоким весом, которые могут привести к высокой вероятности ошибочного декодирования ТК. На рис. 2.6 приведены структуры информационных ошибок, связанных с уровнем (весом) ошибок блокового перемежителя.

а



б



в

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

Рис. 2.5. Общие ограниченные кодовые слова блокового перемежителя и турбокода с порождающим номиналом G = {23;37}:
а – входные последовательности с информационным весом 4, формирующие общие ограниченные кодовые слова в блоковом перемежителе;
б – входные последовательности с информационным весом 6, формирующие общие ограниченные кодовые слова;
в – входные последовательности с информационным весом 9, формирующие общие ограниченные кодовые слова

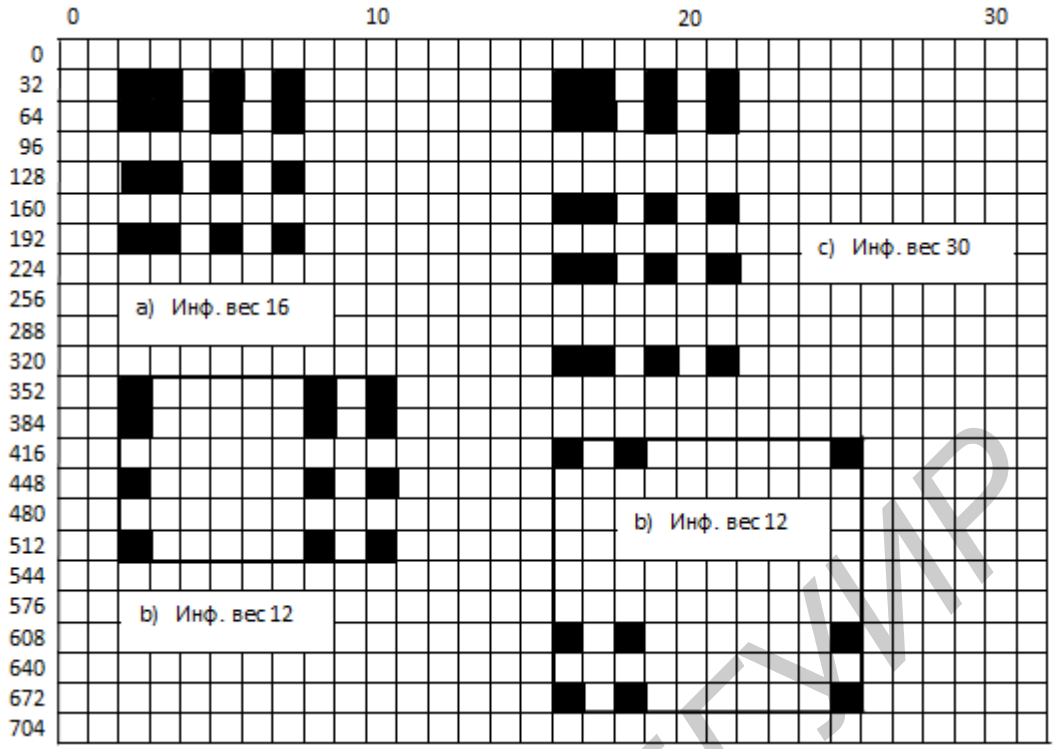


Рис. 2.6. Взаимосвязь ошибочных информационных символов с уровнем ошибок блокового перемежителя

Высокий вес перемеженных последовательностей приводит в среднем для данного блокового перемежителя к 14,86 ошибочным информационным символам на блок данных из 1024 информационных символов. Эти результаты анализа перемежающих свойств блокового перемежителя характеризуют его низкую эффективность в совместном использовании с турбокодами.

Важным параметром деперемежителей/перемежителей является *спектральное расстояние*, характеризующее количество информационных (кодовых) последовательностей с равным кодовым расстоянием и весом α(d,i). В соответствии с [1–9], расчет *спектральных расстояний* блокового перемежителя выполняется простым вычислением линейных смещений ошибок кодовых слов с малым весом внутри перемежителя. В соответствии с рис. 2.5, для данного перемежителя существует 676 линейных смещений ошибок с малым информационным весом d = 16 внутри перемежителя рангом 32×32. Это означает, что при использовании блочного перемежителя вероятность ошибочного декодирования ТК остается достаточно большой. На рис. 2.7 приведены кривые вероятности ошибочного декодирования ТК с R=1/3 с порождающим полиномом G = {23;37}, G = 8 при использовании блочного перемежителя. Алгоритм декодирования ТК Log-MAP, длина информационного блока – 1024 символа и количество состояний кодовой решетки – 16.

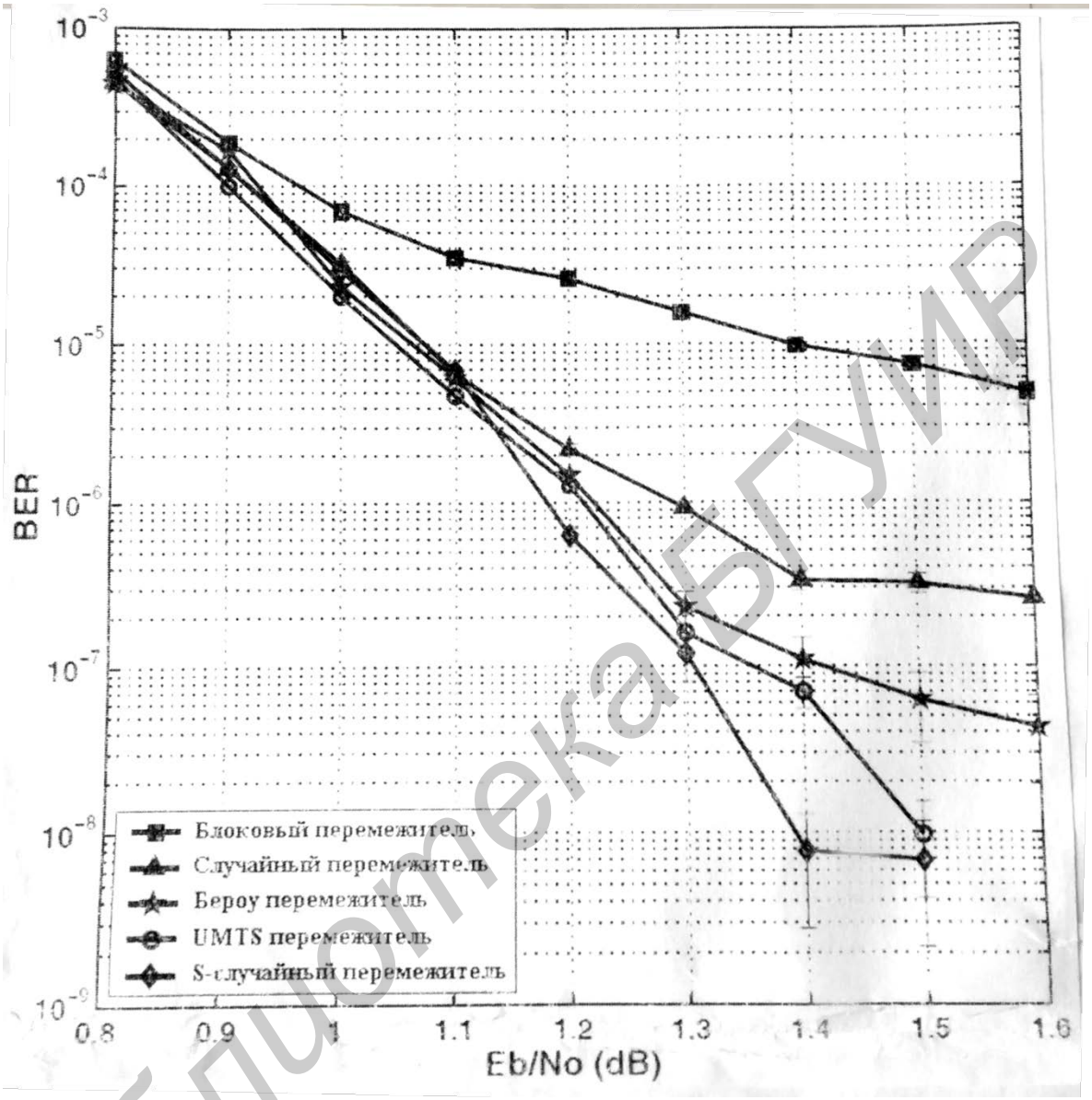


Рис. 2.7. Кривые вероятности ошибочного декодирования ТК с R = 1/3, G = {23;37}, G = 8 при использовании разных типов де/перемежителей

Перемежитель Берроу – Главье

Ученые К. Берроу и А. Главье предложили использовать неоднородные перемежители для ТК [58], чтобы разрушить (разбить) низковесную кодовую структуру, связанную с блоковым перемежением, как это показано на рис. 2.5.

Перемежитель Берроу – Главье полностью определяется количеством строк $n_1 = 2^m$ и количеством столбцов $n_2 = 2^x$, где m и x – целые числа, и

набором простых чисел ∫{l}, ℓ = 1…8: эти значения параметров включают в себя псевдослучайные элементы. В [19, 23, 27] было выполнено моделирование вероятности ошибочного декодирования ТК при использовании де/перемежителя Берроу – Главье с различными наборами ∫{l}, l = 1…8 простых чисел.

Для каждой входной позиции символов i выходная позиция ε(i) описывается следующим уравнением [11, 29]:

$$\varepsilon(i) = c(i) + c \cdot z(i), \tag{2.6}$$

где $c(i) = \left(\left(\frac{c}{2} + 1\right) \cdot (z_0 + c_0)\right) \mathrm{mod} C;$

$z(i) = (\rho(l+1)(c_0 + 1) - 1) \mathrm{mod} R;$

$z_0 = i \mathrm{mod} C, \quad c_0 = \frac{(1 - z_0)}{C}, \quad l = (z_0 + c_0) \mathrm{mod} X, \quad x = 8.$

Первые две строки матрицы транспонируются, третья строка выбирает простое число из серии ∫{ρ(l)} в соответствии с транспонированной строкой и столбцом двоичного разряда. Четвертая строка выбирает новую строку в зависимости от выбранного простого числа. Символы расположены в восьми позициях друг от друга, когда входы в перемежитель находятся на той же новой строке. Пятая строка выбирает новый столбец, так что символы в соседних столбцах при входе в перемежитель располагаются в c/2 + 1 столбцах, кроме случая выхода из перемежителя.

В табл. 2.2 и 2.3 приведены соответственно ошибочные кодовые последовательности и значения спектральных расстояний перемежителя Берроу – Главье, полученные путем моделирования информационных последовательностей с различными весами [18].

Таблица 2.2

Ошибочные последовательности данных перемежителя Берроу – Главье

Стандартная последовательность	Перемеженная последовательность	Кодовый вес
22 блока данных с информационным весом 2		
3 × (245,260)		
3 × (402,417)	-> (194,209)	22
2 × (146,161)	-> (773,788)	22
	-> (517,532)	22

Примечание. 22 ошибочных блока с содержанием 44 битовых ошибок. Среднее значение 2,00 битовых ошибок на ошибочный блок.

Таблица 2.3

Спектральные расстояния перемежителя Берроу – Главье

Информационный вес	Ошибки информационных бит
I = 1	Ошибки с малым весом (< 50) не найдены
I = 2	11 последовательностей с кодовым весом 22: одинаковые (146,161) -> (517,532) (178,193) -> (937,952 (245,260) -> (194,205)
I = 3	1 последовательность с кодовым весом: (595,603,616) -> (748,763,1011)
4 < i < 6	Ошибки с малым весом (< 50) не найдены

Примечание. Минимальный найденный кодовый вес $d_{min} = 22$

Моделированием было установлено [1–8], что перемежитель Берроу – Главье не может гарантировать, что пары символов, расположенных на расстоянии 15 символов друг от друга, не будут перемежены в пары ошибок, также расположенных на расстоянии 15 символов друг от друга и, следовательно, такая закономерность будет сохраняться для других вариантов расположенных пар позиций ошибочных информационных символов. На рис. 2.7 приведена кривая вероятности ошибочного декодирования ТК при использовании перемежителя Берроу – Главье, которая показывает, что вероятность ошибочного декодирования ТК с данным перемежителем информационных символов меньше, чем при использовании блокового перемежителя.

Случайный перемежитель

Случайный перемежитель – это перемежитель со случайным соответствием между входными и выходными позициями передаваемых информационных символов. То есть для перемежителя длины N входная информационная последовательность символов i = 0…(N–1). Преимуществом случайных перемежителей является их простота реализации, а недостатком – невозможно гарантировать минимально рассеивающую (перемеженную) характеристику и уровень ошибок перемежения. Случайные перемежители, как правило, имеют очень низкий s-*параметр* и очень высокую *дисперсию* и в этом плане они являются «противоположными» блоковым перемежителям.

В табл. 2.4 приведены ошибочные информационные последовательности, полученные моделированием случайного перемежителя. В круглых скобках указывается длина информационной последовательности, а множитель перед

последовательностью указывает на количество раз, когда отдельная последовательность была ошибочной при моделировании. Символ → обозначает операцию перемежения (перемешивания). Буква «t» перед двоичным разрядом означает, что турбокодер вставил этот бит для перевода матрицы RSC_1-кодера в нулевое состояние. Это означает, что информационная последовательность не самоограничена. Такие последовательности, как правило, имеют высокий вес.

Таблица 2.4

Ошибочные последовательности данных случайного перемежителя

Стандартная последовательность	Перемеженная последовательность	Кодовый вес
20 блоков данных с информационным весом 4		
10 × (893,909,910,915)	-> (462,466,467,474)	22
4 × (115,376,378,385, t1021, t1023)	-> (69,169937,938,946,950)	999
10 блоков данных с информационным весом 3	-> (418,425,431)	25
2 × (712,717,737)		
2 × (613,621,629,t1020,t1022)	->(371,449,451,458,926)	525
6 блоков данных с информационным весом 2		
4 × (781,796)	-> (683,696)	22
2 × (468,483)	-> (789,804)	22
5 блоков данных с информационным весом 5		
1 × (967,969,980,982,983)	-> (589,597,604,624,628)	37

Примечание. 60 блоков с 526 битовыми ошибками. Среднее значение 8,77 битовых ошибок на ошибочный блок.

Из табл. 2.4 следует, что случайный перемежитель может формировать (генерировать) последовательность с минимальным весом 2 и минимальным кодовым весом 22 (d_{min} = 22). Однако случайный характер перемежения может сформировать кодовое слово с большим весом (d_{min} = 999 и 525); вероятность появления таких кодовых слов маловероятна. Меньшая кратность кодовых слов при d_{min} приводит к снижению количества ошибок в информационных и кодовых последовательностях, что иллюстрируется кривой вероятности ошибочного декодирования ТК (см. рис. 2.7). Среднее значение ошибочных символов на блок равно 8,77.

На рис. 2.8 приведены гистограммы спектральных расстояний случайного перемежителя, а в табл. 2.5 – для s-случайного перемежителя [11, 19, 29].

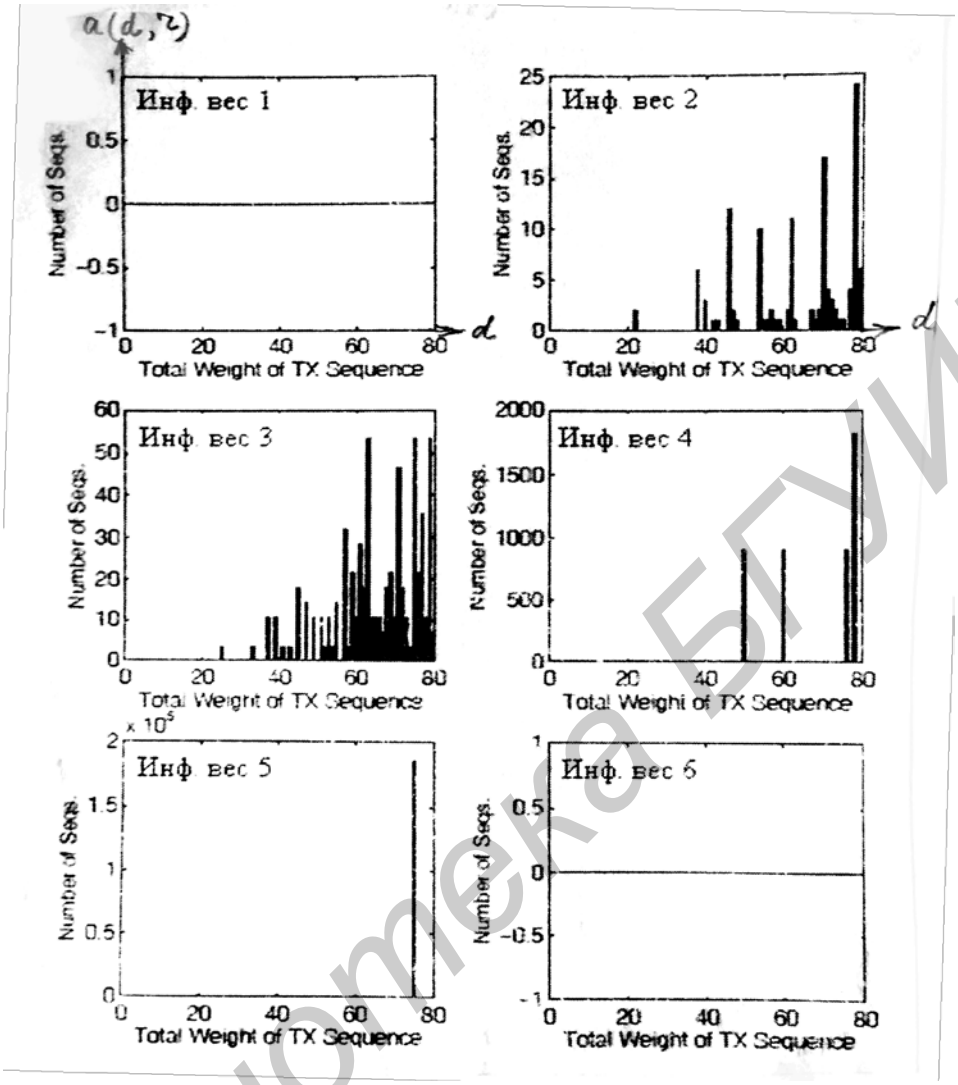


Рис. 2.8. Спектральные расстояния α(d;r) = 1…6, d = 1…80 случайного перемежителя

Анализированы последовательности с малым кодовым весом. Ось абсцисс «x» каждого графика представляет кодовый вес d, а ось ординат «y» представляет *спектральное расстояние* α(d;r). Приведенные гистограммы и табличные данные позволяют выбрать оптимальные параметры случайного и s-случайного перемежителя, обеспечивающие наилучший процесс перемежения информационных символов. Так, например, s-случайный перемежитель перемежает (рассеивает) информационный вес двух последовательностей с кодовым весом d = 22. Это вызвано тем, что анализируемый s-случайный перемежитель имеет s-параметр, равный 19 (s = 19). Пары символов данных,

разделенные 15-ю битовыми (символьными) позициями, должны передаваться как минимум 19-ю битовыми позициями. Такая чередованная пара будет перемеженной с интервалом в 15 символов (бит).

Таблица 2.5

Спектральные расстояния для s-случайного перемежителя

Информационный вес	Модели данных	
1	Не найдены модели данных с низким весом (≤ 50)	
2	Кодовый вес 29	1 Последовательность: (996,998,t1020) → (317,319,341)
	Кодовый вес 30	7 Последовательность: (71,86) → (424,454) (108,123) → (670,700) (301,316) → (597,627)
3	Кодовый вес 31	1 Последовательность: (281,289,302) → (32,39,60)
	Кодовый вес 32	1 Последовательность: (55,73,74) → (720,735,1016)

Примечание. Минимальный определенный кодовый вес $d_{min} = 29$

Псевдослучайный перемежитель

Псевдослучайный перемежитель является вариацией случайного перемежителя, который построен таким образом, чтобы обеспечивать высокий s-*параметр* s. Перемежитель был впервые описан в [23, 28]. Для каждой следующей входной позиции i выходная позиция ε(i) выбирается случайно. Это значение сравнивается с s заранее выбранными выходными позициями. Если любая из этих s позиций находится на расстоянии s от выбранной в данный момент позиции, то текущая позиция отбрасывается. Этот процесс повторяется, пока не выбраны все N позиции перемежителя. Итак, это перемежитель, который в основе имеет случайную структуру, но гарантирует минимальное распространение ошибочных пар. Время, затраченное на поиск новой подходящей позиции, непропорционально увеличивается с увеличением s, а методика не всегда позволяет найти позицию, которая удовлетворяет определенным значения s. Как показано в [23], при выборе $s < \sqrt{N/2}$ обычно находится решение в разумные сроки. Это практическое значение s было приведено, чтобы сократить значение $\sqrt{N/2}$ с ростом размера перемежителя.

Псевдослучайные перемежители работают очень хорошо (см. рис. 2.7), имеют один из лучших показателей уровня ошибок среди перемежителей, рассмотренных здесь. Тем не менее ключевым недостатком этого перемежителя

является долговременность метода поиска, описанного ранее. Как известно, в сетях передачи данных, которые обрабатывают пакеты разной длины, может возникнуть необходимость производить перемешивание «на лету» перед применением прямого исправления ошибок в пакете. Это может занять непозволительно долгое время в случае псевдослучайного перемежения. Другим вариантом было бы хранить преобразования для пакетов различной длины. Это, однако, займет большее количество памяти.

UMTS-перемежитель

UMTS-перемежитель был разработан для мобильных систем связи 3-го поколения (3GPP-3rd Generation Partnership Project – проект партнерства 3-го поколения) [21, 24, 25]. Турбодекодер использует два RSC с k = 2 и ϑ = 3, т. е. 8-позиционные RSC для обеспечения допустимой вероятности ошибок по символам (битам) не более 10^{-6}. Деперемежитель/перемежитель выполняется в виде процессора и процесс перемежения информационных символов выполняется в три этапа. Сущность этапов перемежения информационных символов состоит в следующем: *на первом этапе* входная информационная последовательность символов записывается в прямоугольную матрицу строка за строкой. Допустимы только некоторые размеры матрицы: конкретные размеры матрицы определяет простое число p, которое используется на втором этапе. Перемежитель может иметь либо 10, либо 20 строк. *Второй этап* – внутристроковая перестановка, т. е. позиции бит перераспределяются в каждой строке. Для данного этапа нужна таблица соответствия (преобразования), чтобы определить преобразованный корень, основанный на значении p, и хранить множество простых чисел, которые не являются сомножителями p – 1. Преобразованный корень и последовательность простых чисел используется для выполнения двухступенчатой внутристроковой перестановки. *Третий этап* – междурядная перестановка – строки сами перераспределяются. Данный этап также требует наличие справочной таблицы для хранения трех возможных моделей междурядной перестановки, обеспечивающих хорошее распространение (распределение, разнесение) для различных длин перемежителя. Наконец, биты считываются столбец за столбцом. Полная информация о реализации UMTS-перемежителя предлагается в технической спецификации 3GPP [21].

Главное преимущество алгоритма UMTS-перемежения заключается в том, что он может создавать детерминированным образом хорошее распространение ошибочной комбинации для перемежителя длиной от 320 до

5114 бит – это хорошая характеристика, например, для гибридных сетей передачи голоса/данных. Результатом является то, что хорошие UMTS-перемежители могут быть сгенерированы «на лету», что устраняет необходимость хранения нескольких преобразований. UMTS-перемежитель имеет частоту появления ошибочных бит, сопоставимую с псевдослучайным перемежителем (см. рис. 2.7). Тем не менее алгоритм UMTS не будет работать для всех длин блоков. Для всех, кроме двух конкретных длин блока, число строк должно быть 20, поэтому ближайшей к стандарту в 1024 бит является длина перемежителя 52×20 = 1040 бит. Система UMTS требует, чтобы 134 «материнских» моделей перемежения хранились таким образом, чтобы пакеты различной длины могли быть закодированы. Эта сложность может помешать его реализации в некоторых системах.

В табл. 2.6 приведены параметры ошибочных информационных последовательностей, полученные путем моделирования работы UMTS-перемежителя на этапе его разработки [21, 24, 25].

Таблица 2.6

Параметры ошибочных информационных последовательностей UMTS-перемежителя

Стандартная последовательность	Перемеженная последовательность	Кодовый вес
2 блока данных с информационным весом 3		
1 × (475,477,484)	-> (881,901,921)	31
1 × (710,732,738)	-> (182,192,202)	33
1 блок данных с информационным весом 4		
1 × (69,70,75,83)	-> (715,735,755,1035)	39

Примечание. 3 ошибочных блока с содержанием 7 битовых ошибок. Среднее значение 2,33 битовых ошибок на ошибочный блок.

Из табл. 2.6 следует, что перемеженные информационные последовательности имеют более высокий минимальный кодовый вес $d_{min} = 31$ по сравнению со *случайным* и *Берроу – Главье перемежителями* и намного меньше, чем *блокового перемежителя*. Также наблюдается отсутствие последовательностей с информационным весом 2.

В табл. 2.7 приведены значения спектрального расстояния UMTS-перемежителя для информационных последовательностей с весом $i = 1…6$ [21, 25].

Результаты моделирования, приведенные в табл. 2.7, показывают, что спектральные расстояния UMTS-перемежителя схожи с результатами s-перемежителя. При информационном весе i = 1 информационные последовательности с малым кодовым весом не обнаружены и при информационном весе i = 2 обнаружена последовательность с кодовым весом 28.

Таблица 2.7

Спектральные расстояния UMTS-перемежителя

Информационный вес	Ошибки информационных бит
I = 1	Ошибки с малым весом (< 50) не найдены
I = 2	Последовательность с кодовым весом 28: (432,447) -> (1011,1031)
I = 3	Последовательность с кодовым весом 29: (88,92,106) -> (55,65,75) Последовательность с кодовым весом 31: (475,477,484) -> (881,901,921) (712,720,733) -> (592,602,612) (748,751,752) -> (322,342,362)
4 < i < 6	Ошибки с малым весом (< 50) не найдены

Примечание. Минимальный найденный кодовый вес $d_{min} = 28$

Перемежитель «строка за строкой», или построчный перемежитель

Пермежитель «строка за строкой» является модификацией блокового перемежителя [20–25]. Сущность модификации блокового перемежителя, содержащего n_1 строк, n_2 столбцов ($n_1 \leq n_2$), состоит в следующем:

– информационные символы записываются последовательно в строку за строкой;

– множество целых чисел $S(r) = S_1,S_2\ldots,S_i$ присваиваются группам M строк в перемежителе в циклической форме, в которой номер строки является фактором n_1, а $S_1,S_2\ldots,S_i$ – простые числа относительно n_2;

– позиции символов (бит) каждой строки кодируются по формуле

$$n_{2new} = (s\{r \bmod M\} \cdot C) \bmod n_2, \qquad (2.7)$$

где *r* – номер строки;

C – номер столбца;

n_{2new} – перемеженный (переставленный) столбец.

Далее информационные символы (биты) считываются по столбцам один за другим.

На рис. 2.9 приведен фрагмент реализации перемежителя «строка за строкой». Для внутристрочной скремблированной последовательности принято s(r) = {1,3}. Эта последовательность представляет смешение альтернативных строк. *Дисперсия* данного перемежителя равна S = 0,032, что почти на порядок больше, чем у бокового перемежителя, но и на порядок меньше, чем у случайного перемежителя. S-параметр для данного перемежителя равен 12, что значительно меньше, чем у блокового перемежителя ($S_б$ = 31) с матрицей размерностью 32×32.

Уменьшение *s-параметра* данного перемежителя по сравнению с блочным перемежителем связано с наличием ошибочных символов в соседних строках, расположенных в одном столбце и при изменении одной пары символов. Это иллюстрирует карта ошибок, приведенная на рис. 2.10, для одного из шаблонов с ограничивающим параметром S = 12.

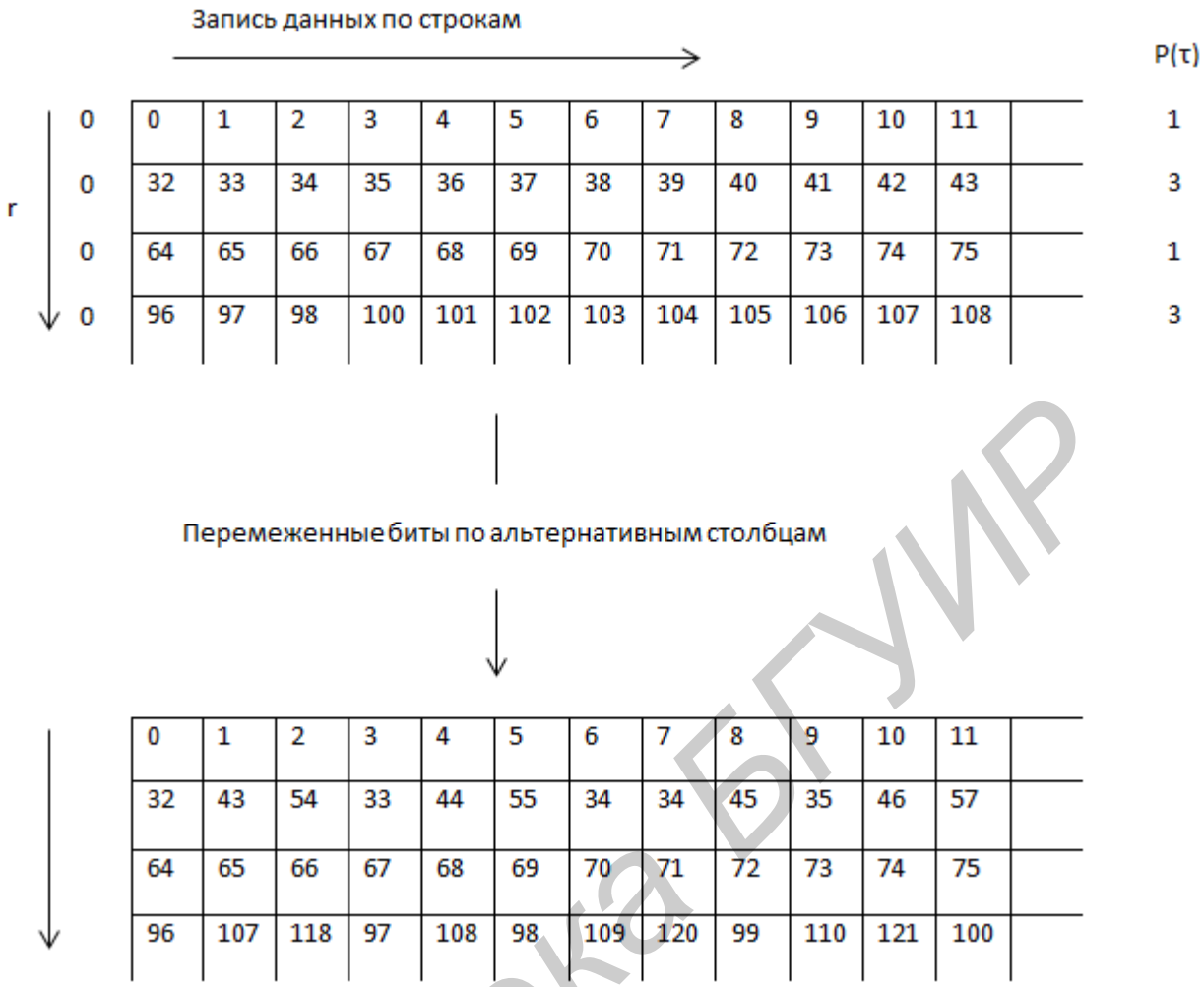


Рис. 2.9. Фрагмент перемежителя «строка за строкой» рангом 32×32 и N = 1024 символа

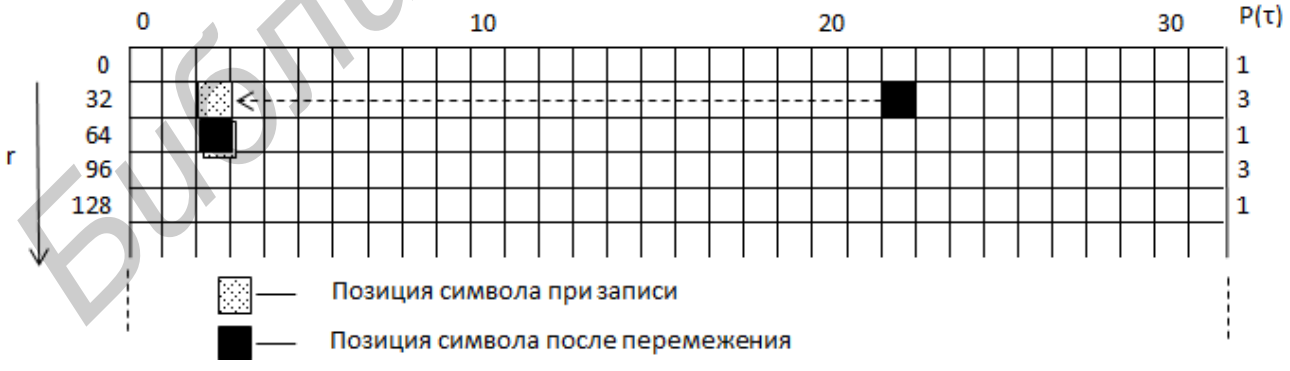


Рис. 2.10. Ошибочные позиции символов, приводящие к уменьшению s-параметра перемежителя «строка за строкой»

Меньшие значения s{l} приводят к более высокому значению s-параметра. В [27–29] показано, что при s(r) = {1,5} s-параметр равен 8, а также, что перемежитель «строка за строкой» перемежает почти все последовательности с заданным значением s-перемежителя, информационный вес которого не превышает 9, т. е. d = 9; расстояние между ошибочными строками таково, что они либо не переставляются вообще (s{l} = 1), либо переставляются одинаково (s{l} = 3). В этом случае позиции символов совпадают с непереставленными строками. С увеличением длины внутристрочной скремблированной последовательности s(r) = {3,7,1,5} перемежающие свойства данного перемежителя улучшаются: все последовательности с низким кодовым весом перемежаются в последовательности с высоким кодовым весом, что приводит к уменьшению вероятности ошибочного декодирования ТК.

Перемежитель «четыре на четыре»

С целью повышения перемежающих свойств перемежителя «строка за строкой» l [21–25] было предложено матрицу перемежителя разделить на фрагменты, размером четыре на четыре, или 4×4. Информационные символы записываются, как и в предыдущем перемежителе строка за строкой. Позиции символов (бит) перемежителя перемежаются. Считывание символов (бит) производится по столбцам.

Алгоритм работы фрагменты (блока) 4×4 перемежителя состоит в следующем:

– перемежитель должен содержать 4i строки и 4i столбцов, где i – целое положительное число;

– символы (биты) записываются (заносятся) в перемежитель по строкам;

– перемежитель разделяется (разбивается) на фрагменты (блоки) рангом 4×4 соответственно строк и столбцов;

– каждый блок 4×4 перемежается (шифруется) по следующей матрице:

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

В общем случае, как отмечают авторы разработки данного перемежителя, необходимо, чтобы группы символов, которые формируют *самоограниченную*

последовательность, не образовывали бы *самоограниченные последовательности*, когда они считываются. На рис. 2.11 представлен процесс записи, перемежения и считывания информационных символов при расположении пары позиций символов в рамках одного фрагмента 4×4 на 30 позиций друг от друга. Из рис. 2.11 видно, что при считывании символов они уже находятся на позициях, отличных от 30. Перемеженные символы всего перемежителся считываются столбец за столбцом.

S-параметр данного перемежителя равен 29, а *дисперсия* ξ = 0,022, что обеспечивает вероятность ошибочного декодирования ТК на уровне использования ТК со случайным перемежителм. Кроме того, данный перемежитель чувствителен к структуре передаваемых информационных последовательностей. Минимальная вероятность ошибочного декодирования ТК ($P_{ош.дек} \approx 10^{-6}$) с данным перемежителем обеспечивается в канале связи с АБГШ при $E_в/N_0$ = 1,5 дБ.

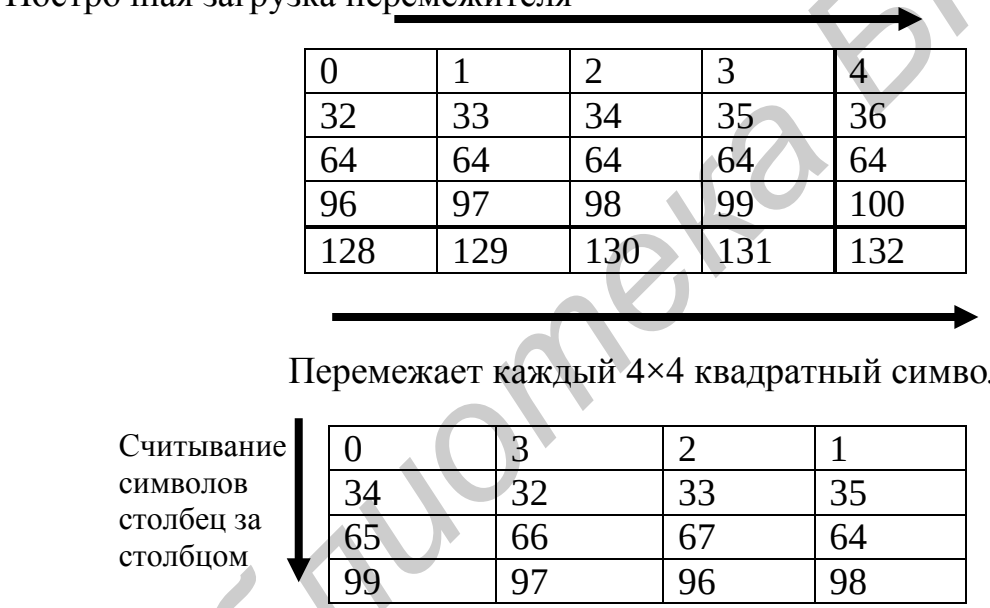


Рис. 2.11. Процессы организации записи перемежения исчитывания информационных символов

York-перемежитель

York-перемежитель является модификацией перемежителя «четыре на четыре» [23–29] и предназначен для обеспечения высокого значения *s-параметра*. Это можно обеспечить, если символ в строке блока 4×4 не перемежается в тот же столбец. Алгоритм работы York-перемежителя состоит в следующем:

– перемежитель должен иметь $n_1 = 4i$ строк и $n_2 = 4i$ столбцов, где i – целое число;

– передаваемые символы записываются построчно по всей матрице перемежителя;

– символы всего перемежителя разбиваются на блоки 4×4, как и в предыдущем перемежителе, а далее осуществляется кодирование (шифрование) символов;

– первый символ в первой строке каждого блока присваивается столбцом к строкам в блоке в произвольном порядке;

– второму символу в первой строке присваивается столбец (колонка), который не был присвоен символу первой строки; строка выбирается наугад. Строка и столбец выбираются случайным образом и необязательно должны отличаться от ранее записанных строк и столбцов;

– этот процесс продолжается до тех пор, пока все символы в первой строке не займут все позиции в блоке. Все символы первой сроки должны теперь быть в различных столбцах;

– три других строки кодируются аналогичным способом, пока не обеспечится их расположение;

– далее символы считываются со всего перемежителя по столбцам.

Рассмотренный процесс перемежения представлен на рис. 2.12.

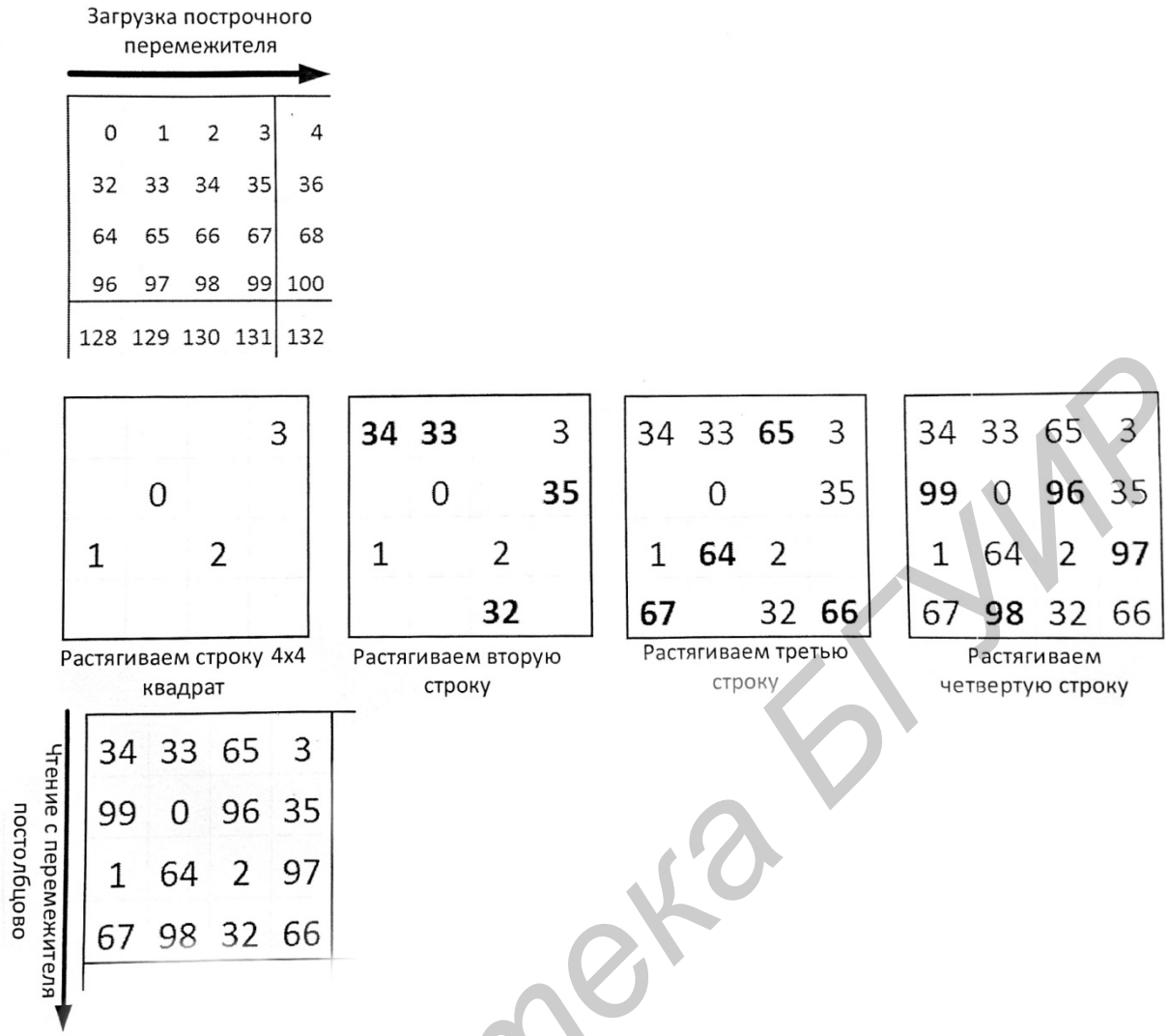


Рис. 2.12. Этапы перемежения информационных символов York-перемежителя

Установлено, что *s-параметр* и *дисперсия* данного перемежителя равны S = 26 и ξ = 0,232, что существенно больше, чем у всех ранее рассмотренных перемежителей. На рис. 2.13 приведены кривые вероятностей ошибочного декодирования ТК при использовании алгоритма декодирования Log-MAP, информационной последовательности длиной L=1024 символов, RSC со скоростью передачи кода R=1/3, 16-позиционной кодовой решетки при наличии в канале связи АБГШ и при использовании перемежителей всех рассмотренных конструкций.

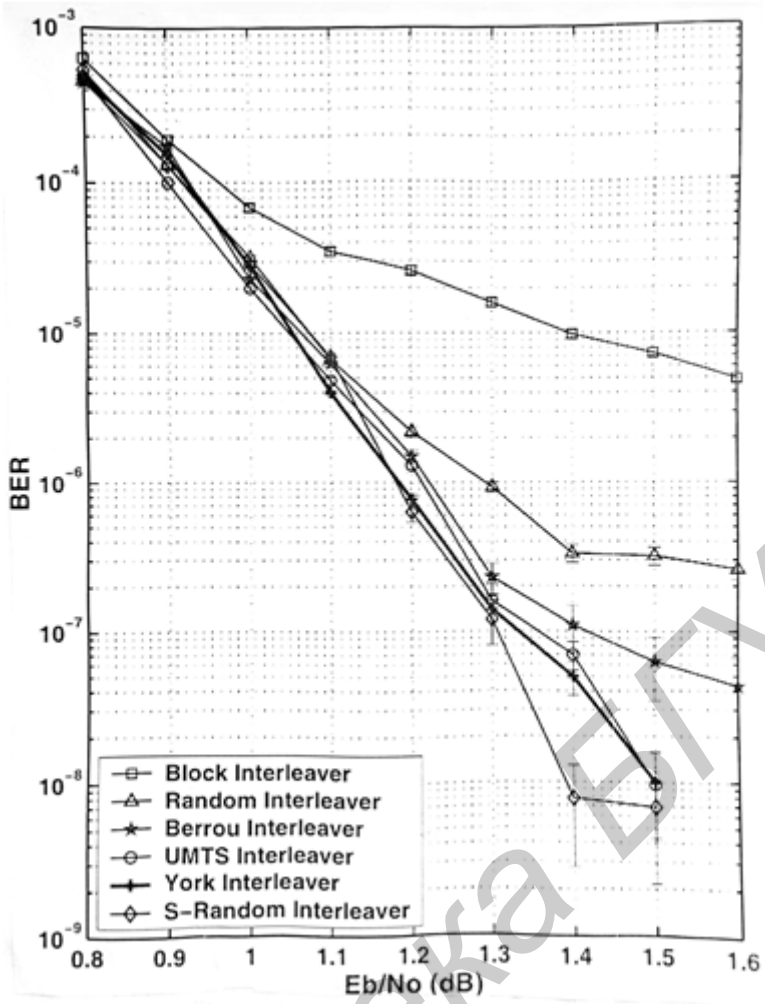


Рис. 2.13. Вероятности ошибочного декодирования ТК при использовании деперемежителей в канале связи с АБГШ при Q = 8 итераций

Для оценки эффективности эксплуатационных характеристик и выбора типа деперемежителя/перемежителя используются три параметра, а именно: *s-параметр*, *дисперсия* и *минимальное расстояние,* или *кодовый вес*. На рис. 2.14 приведены гистограммы спектральных расстояний рассмотренных деперемежителей/перемежителей [23–27].

Сумма спектральных расстояний для информационных последовательностей с весом i = 2; 3 и 4 определялась как $\alpha(d) = \alpha(d,2) + \alpha(d,3) + \alpha(d,4)$. Ось «y» на гистограмме ограничена $\alpha(d) = 60$ последовательностями, типы деперемежителей/перемежителей упорядочены в зависимости от вероятности ошибочного декодирования. Из приведенных гистограмм и кривых вероятностей ошибочного декодирования ТК (см. рис. 2.7) следует, что наименьшая вероятность ошибочного декодирования обеспечивается при использовании s-случайного перемежителя. Следовательно, «хороший» деперемежитель/перемежитель должен иметь высокий минимальный кодовый

вес или большое минимальное расстояние d_{min}, малое значение спектрального расстояния α(d;i), высокую дисперсию и высокий s-*параметр*, а также высокую скорость записи и считывания информационных символов (бит).

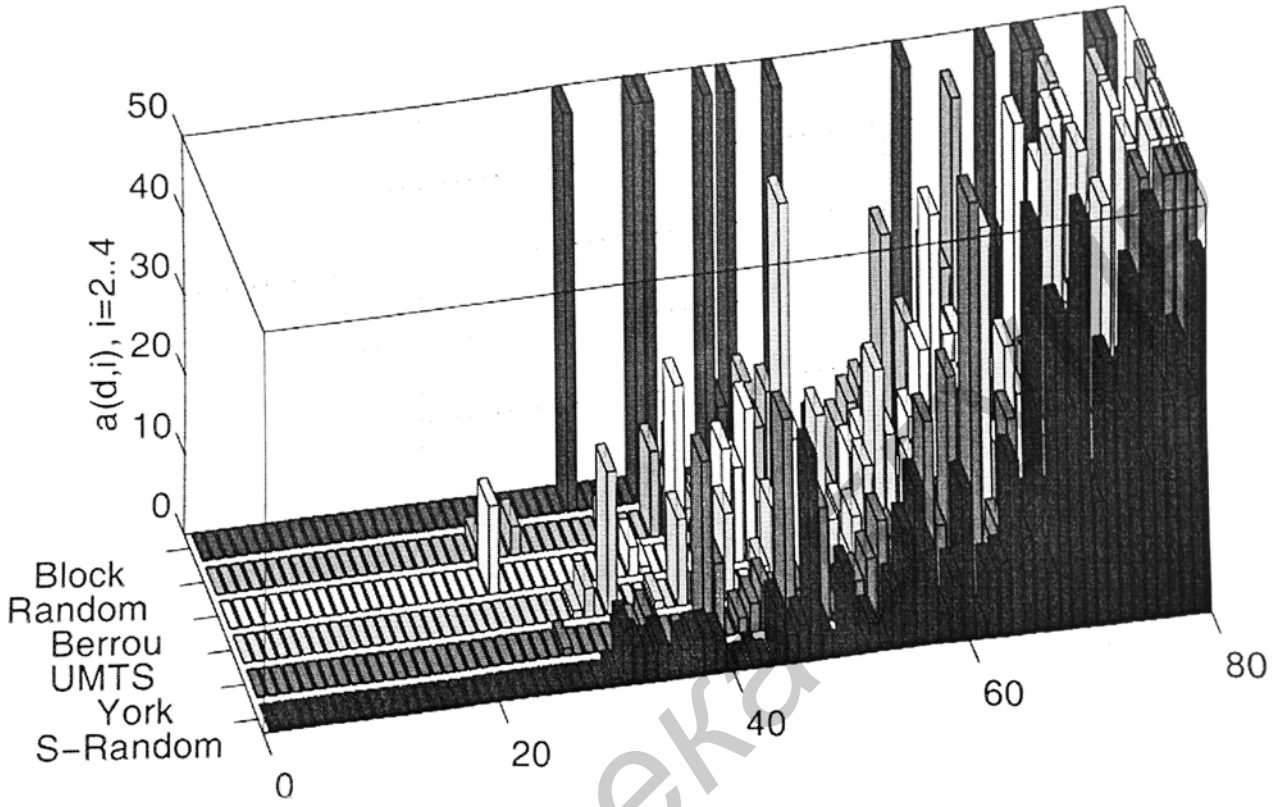


Рис. 2.14. Спектральные расстояния перемежителей, состоящие из суммы расстояний α(d;i) при информационном весе I = 2; 3 и 4 последовательностей

ЗАКЛЮЧЕНИЕ

Из представленного материала следует, что ТК обеспечивает высокую достоверность передачи информации ($P_{ош.пр} \leq 10^{-6}$) при низких отношениях $P_с/P_ш$ = 1,5–2,0 дБ, гарантируя, таким образом, эффективное использование пропускной способности каналов связи. Установлено, что турбокодирование наиболее эффективно при совместном использовании с многопозиционными методами модуляции ФМ и АМ. Однако проблемными остаются вопросы выбора типа перемежителя, количества компонентных кодов, их тип, а также уменьшение задержки информации (данных) при декодировании кодовых последовательностей.

ПЕРЕЧЕНЬ УСЛОВНЫХ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

Сокращения

АВ – алгоритм Витерби

АС – анализатор синдрома

ДКС – дискретный канал связи

ДСК – двоичный симметричный канал

Декодер – декодирующее устройство

ДШС – дешифратор синдрома

КАМ – квадратурная амплитудная модуляция

КРВ-1/k_0 – коммутатор распределения ветвей одного потока двоичных символов на $k_0 (k_0 \geq 2)$ подпотоков

КРВ-1/n_0 – коммутатор распределения ветвей одного потока двоичных символов на $n_0 (n_0 = k_0 + 1)$ подпотоков

КОВ-k_0/1 – коммутатор объединения k_0 ветвей двоичных символов в один поток

КОВ-n_0/1 – коммутатор объединения n_0 ветвей двоичных символов в один поток

КО – корректор ошибок

Кодер – кодирующее устройство

ЛЭ – линейный элемент

МЭ – мажоритарный элемент

МФМ – многократная фазовая модуляция (манипуляция)

ПД – пороговое декодирование

ПИ – передача информации

ПЭ – пороговый элемент

СК – сверточный код

ССК – самоортогональный СК

СБЛК – систематический блоковый линейный код

ТК – турбокод

$ФПС_{к/д}$ – формирователь проверочных символов (к – кодера, д – декодера)

ФСС – формирователь синдромных символов

ЦК – циклический код

ЭВК – энергетический выигрыш кодирования

ЭВМ – энергетический выигрыш модуляции

BER (Bit-Error-Rate) – частота ошибочных бит, или вероятность ошибочного приема бита (двоичного символа)

BPSK (BinaryPhaseShiftKeying) – бинарная (двоичная) фазовая манипуляция

CDMA (CodeDivisionMultipleAccess) – мультиплексирование с кодовым разделением каналов

DVB (DigitalVideoBroadcasting) – цифровое телевидение

GSM (GlobalSystemforMobileCommunication) – глобальная система мобильных телекоммуникаций

LLR (Log-LikelihoodRatio) – логарифмические отношения правдоподобия

RSC (RecursiveSystematicCode) – рекурсивный систематический код

SISO (SoftInputSoftOutput) – мягкий вход/мягкий выход

SNR (Signal Noise Ratio) – отношение сигнал/шум

SOVA (Soft Output Viterbi Algorithm) – алгоритм Витерби

UMTS (Universal Mobile Telecommunications System) – универсальная мобильная телекоммуникационная система

P(x) или g(D) – порождающий полином

h(x) – проверочный полином

Обозначения

d – кодовое расстояние

d_0 – минимальное кодовое расстояние

$d_{св}$ – свободное кодовое расстояние

R = k/n – скорость передачи блокового кода: k – количество информационных символов, n = k + l – количество кодовых символов, l – количество контрольных (проверочных) символов

R = (1 – R) · 100 % – относительная избыточность кода

$R = k_0/n_0$ – скорость передачи сверточного кода: k_0-мини-блок информационных символов, n_0-мини-блок кодовых символов

$t_{исп} \leq (d_0 - 1)/2$ – кратность (количество) исправляемых ошибочных двоичных символов

$t_{обн} \leq d_0 - 1$ – кратность (количество) обнаруживаемых ошибочных двоичных символов

$\vartheta = k + 1$ – длина кодового ограничения сверточного кода: k – количество ячеек памяти регистра сдвига кодера

$P_с/P_ш$ – отношение мощности сигнала к мощности шумов

$E_в/N_0$ – отношение энергии сигнала, необходимое для передачи одного бита информации, к спектральной плотности шумов

$P_к$ – вероятность ошибочного приема двоичного символа (бита) на выходе демодулятора дискретного канала связи

$P_{ош.дек}$ – вероятность ошибочного декодирования двоичного символа (бита)

ЛИТЕРАТУРА

1. Andrews, I. D. Seleetion of component codes for turbo coding based on convergence propezties / I. D. Andrews // Annales des Telecommunications. Special issue on turbo codes. – 1999. – Vol. 54, № 3–4.
2. Berrou, C. Near Shannon Turbo Codes / C. Berrou, A. Glavieux, P. Thitimajshima // Proe. Of the Jntern. Conf. on Commun, Geneva, Switzerland, May, 1993. – P. 1064–1070.
3. SACET. Generical Two Dimensional Block Turbo Code Deco-der. Preliminary Product Specification, 15 th. March, 2002.
4. Segners. On the Free Distance of TURBO Codes and Related Product Codes // Einal Report, Diploma Project, 1995, Number 6613, Swiss Federal Institute of Technology, Zurich, Switzerland, August, 1995.
5. Jin, H. Irregular repeat-accumulate codes / H. Jin, A. K. Khandekar, R. McEliece // Proe. 2nd. Int. Symp. On Turbo Codes and Related Topics, Brest, France, 2000, Sept.-Pl.
6. Williams, D. Hurbo Product FEC Contribution / D. Williams // IEEE 802.16.1 pc-00/35. 2000, June 19.
7. Jin, H. Analisys and Desing of Turbo-Like Codes. Ph.D. dissertation / H. Jin. – California, 2003.
8. Robertson, P. A Comparison of Optimal and Sub-Optimal MAP Decoding Algorithms Operating in the Log Domain / P. Robertson, E. Villebrun, P. Höher // Proceedings of the International Conference on Communications, Seattle, Unites States, June 1995. – P. 1009–1013.
9. Robertson, P. Improving Decoder and Code Structure of Parallel Concatenated Recursive Systematic (Turbo) Codes / P. Robertson // IEE Trans, of International Conference on Universal Personal Communications, San Diego, Sept. 1994. – P. 183–187.
10. Hagenauer, J. Iterative (Turbo) Decoding of Systematic Convolutional Codes with the MAP and SOVA Algorithms / J. Hagenauer, P. Robertson, L. Papke // ITG –Fachbericht 130, Oct. 1994. – P. 21–29.
11. Heegard, C. Turbo Coding / C. Heegard, S. B. Wicker. – Kluwer Academic Publishers, 1999.
12. Barbulescu, A. S. On Terminating the Trellis of Turbo–Codes in the Same State / A. S. Barbulescu, S. S. Pietrobon // IEE Electronics Letters. – 1995. – Vol. 31, №1.
13. Blackert, W. J. Turbo Code Termination and Interleaver Conditions / W. J. Blackert, E. K. Hall, S. G. Wilson // IEE Electronics Letters. – 1995. – Vol. 31, №24. – P. 2082–2084.

14. Forney, G. D. The Viterbi Algorithm / G. D. Forney // Proceedings IEEE. – 1973. – Vol. 61, №3. – P. 268–278.
15. Koch, W. Optimum and Sub-Optimum Detection of Coded Data Disturbed by Time Varying ISI / W. Koch, A. Baier // Proceedings GLOBECOM '90, San Diego, Dec. 1990. – P. 1679–1684.
16. Petersen, J. Implementierungsaspekte zur Symbol-by-Symbol MAP Decodierung von Fal-tungscodes / J. Petersen // In ITG-Fachbericht 130, Oct. 1994. – P. 41–48.
17. Robertson, P. Illuminating the Structures of Code and Decoder for Parallel Concatenated Recursive Systematic (Turbo) Codes / P. Robertson // Proceedings of GLOBECOM 94, San Francisco, December 1994. – P. 1298–1303.
18. Divsalar, D. Turbo Codes for PCS Applications / D. Divsalar, F. Pollara // IEEE International Conference on Communications, ICC '95, Seattle, 1995. – P. 54–59.
19. Said, F. Improving the Random Interleaver for Turbo Codes / F. Said, A. Aghvami, W. Chambers // Electronics Letters. – 1999. – Vol. 35, №25. – P. 2194–2195.
20. Yuan, J. Combined Turbo Codes and Interleaver Design / J. Yuan, B. Vucetic, W. Feng // IEEE Transactions on Communications. – 1999. – Vol. 47, №4. – P. 484–487.
21. 3GPP, "Technical Specification, Group Radio Access Network (Multiplexing and Channel Coding for FDD)", Tech. Rep., 3rd Generation Partnership Project, 1999.
22. Takeshita, O. New Deterministic Interleaver Designs for Turbo Codes / O. Takeshita, D. Costello // IEEE Transactions on Information Theory. – 2000. – Vol. 46, №6. – P. 1988–2006.
23. Berrou, C. Near Optimum Error Correcting Coding and Decoding: Turbo-Codes / C. Berrou, A. Glavieux // IEEE Transactions on Communications. – 1996. – Vol. 44, №10. – P. 1261–1271.
24. ETSI TS 125 212 V3.3.0 (2000-06) DTS/TSGR-0125212U Universal Mobile Telecommunications System (UMTS); Multiplexing and channel coding (FDD) [Electronic resource]. – Mode of access: www.etsi.org.
25. ETSI TS 125 222 V3.2.1 (2000-05) RTS/TSGR-0125222UR Universal Mobile Telecommunications System (UMTS); Multiplexing and channel coding (TDD) [Electronic resource]. – Mode of access: www.etsi.org.
26. Dolinar, S. Weight Distributions for Turbo Codes using Random and Non-random Permutations / S. Dolinar, D. Divsalar // Tech. Rep., JPL, August 1995.
27. Pietrobon, S. A Simplification of the Modified Bahl Decoding Algorithm for Systematic Convolutional Codes / S. Pietrobon, A. Barbulescu // International

Symposium on Information Theory and its Applications, Sydney, Australia, Sep. 1994. – P.1073–1077.
28. Robertson, P. Optimal and Sub-Optimal MAP Algorithms Suitable for Turbo Decoding / P. Robertson, P. Hoeher, E. Villebrun // European Transactions on Telecommunications. – 1997. – Vol. 7, №2. – P. 119–125.
29. Berrou, C. Reflections on the Prize Paper: Near optimum error correcting coding and decoding: turbo codes / C. Berrou, A. Glavieux // IEEE Information Theory Society Newsletter. – 1998. – Vol. 48, №2.

Св. план 2014, поз. 59

Учебное издание

Королев Алексей Иванович
Конопелько Валерий Константинович

ТУРБОКОДЫ И ИТЕРАТИВНОЕ ДЕКОДИРОВАНИЕ

УЧЕБНО-МЕТОДИЧЕСКОЕ ПОСОБИЕ

Редактор *Е. И. Герман*
Корректор *Е. Н. Батурчик*
Компьютерная правка, оригинал-макет *В. М. Задоля*

Подписано в печать 29.09.2015. Формат 60x84 1/16. Бумага офсетная. Гарнитура «Таймс».
Отпечатано на ризографе. Усл. печ. л. 4,53. Уч.- изд. л. 4,0. Тираж 50 экз. Заказ 464.

Издатель и полиграфическое исполнение: учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники».
Свидетельство о государственной регистрации издателя, изготовителя,
распространителя печатных изданий №1/238 от 24.03.2014,
№2/113 от 07.04.2014, №3/615 от 07.04.2014.
ЛП №02330/264 от 14.04.2014.
220013, Минск, П. Бровки, 6