Лабораторная работа №

Тема: «Избыточное кодирование данных в информационных системах. Итеративные коды»

Цель: приобретение практических навыков кодирования/декодирования двоичных данных при использовании итеративных кодов

Задачи:

1. Закрепить теоретические знания по использованию итеративных кодов для повышения надежности передачи и хранения в памяти компьютера двоичных данных.

2. Разработать приложение для кодирования/декодирования двоичной информации итеративным кодом с различной относительной избыточностью кодовых слов.

3. Результаты выполнения лабораторной работы оформить в виде описания разработанного приложения, методики выполнения экспериментов с использованием приложения и результатов эксперимента.

1. Теоретические сведения

1.1. Итеративные коды: определение, принципы построения и использования

Итеративные коды относятся к классу *кодов произведения*.

Кодом произведения двух исходных (базовых) помехоустойчивых кодов называется такой *многомерный помехоустойчивый код*, кодовыми последовательностями которого являются все двумерные таблицы со строками кода (k_1) и столбцами кода (k_2).

Итеративные коды могут строиться на основе использования дву-, трехмерных матриц (таблиц) и более высоких размерностей. Каждая из отдельных последовательностей информационных символов кодируется определенным линейным кодом (групповым или циклическим). Получаемый таким образом итеративный код также является *линейным*.

Простейшим из итеративных кодов является *двумерный* код с проверкой на четность по строкам и столбцам. Итеративные коды, иногда называемые *прямоугольными* кодами (англ. *rectangular code*) либо *композиционными* (англ. *product code*), являются одними из самых простых (с точки зрения аппаратной реализации) избыточных кодов, позволяющих исправлять ошибки в информационных словах.

Основное достоинство рассматриваемых кодов – простота как аппаратной, так и программной реализации. Основной недостаток – сравнительно высокая избыточность.

В упомянутой двумерной матрице кодовые слова записываются в виде таблицы. Проверочные символы вычисляются исходя из того, что строки и столбцы должны содержать четное (нечетное) число единиц. Например, при кодировании информационного слова X_k = **011101111** с помощью таблицы с четностью по строкам и столбцам получим избыточные символы $X_r = X_h, X_v, X_{hv}$ = *0010011*, как показано на рис. 5.1 (информационные символы выделены жирным шрифтом, а проверочные – курсивом).

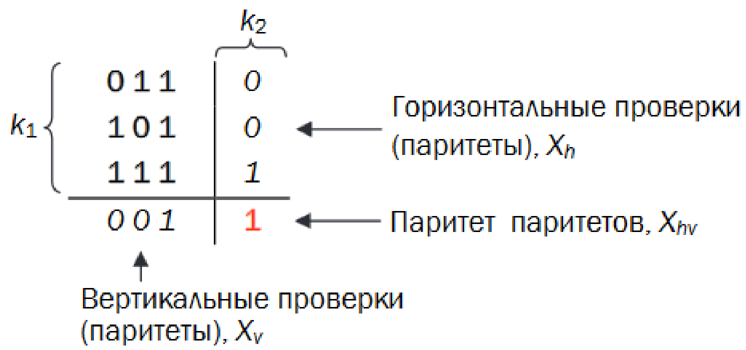

Рис. 5.1. Пояснение к принципу формирования избыточных символов итеративного кода

В соответствии с рис. 5.1 кодовое слово будет иметь следующий вид: X_n = **011101111***0010011*. Как видно, избыточные символы (называемые также паритетами) в приведенном кодовом слове в принятом порядке (X_h, X_v, X_{hv}) записываются сверху вниз, справа налево. Возможен обратный или иной порядок. Важно только, чтобы при декодировании сообщения использовался аналогичный порядок следования паритетов. Символ X_{hv} (паритет паритетов) равен сумме по модулю 2 символов информационного слова X_k, а также проверочных символов X_v и X_h.

Поскольку двумерная матрица формируется как комбинация двух кодов простой четности (по каждому измерению), каждый из которых характеризуется минимальным кодовым расстоянием $d_{\min} = 2$, то полученный итеративный код ($r = k_1 + k_2$) будет характеризоваться минимальным кодовым расстоянием, равным произведению $d_{\min}$ по строкам и по столбцам, т. е. 4.

Использование символа X_{hv} обеспечивает минимальное кодовое расстояние такого итеративного кода $d_{\min}$ ($r = k_1 + k_2 + 1$) на единицу больше. В этом легко обнаруживается сходство кода с кодом Хемминга при $d_{\min} = 4$.

Нетрудно также представить процесс вычисления проверочных символов кодового слова для примера на рис. 5.1 с помощью проверочной матрицы Хемминга и соотношения (4.4). Для указанного примера проверочная матрица кода с $d_{\min} = 3$ выглядит так:

$$H_{15,9} = \begin{array}{ccccccccccccccc}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & & & & & \\
0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & & 1 & & & & \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & & & 1 & & & \\
1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & & & & 1 & & \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & & & & & 1 & \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & & & & & & 1
\end{array}$$

Передачу символов кода обычно осуществляют последовательно символ за символом, от одной строки к другой, либо параллельно целыми строками.

Как показано на рис. 5.1, проверочный символ есть сверка по модулю 2 информационных символов, записанных в соответствующие строку или столбец матрицы.

Декодирование начинают сразу, не ожидая поступления всего блока информации. Проверка соответствия избыточных символов полученного слова ($Y_r = Y_h$, Y_v, Y_{hv} либо $Y_r = Y_h$, Y_v) при декодировании позволяет обнаружить любое нечетное число искаженных символов, расположенных в одной строке или в одном столбце. Формально такое декодирование осуществляется сравнением принятых (Y_h, Y_v, Y_{hv}) и вновь вычисленных (Y'_h, Y'_v, Y'_{hv}) для полученного слова паритетов. В упрощенной форме это показано на рис. 5.2. Определение местоположения одиночной ошибки по строке указывает на наличие ошибки в этой строке матрицы, а проверка по столбцу – конкретный символ (рис. 5.2, *а*).

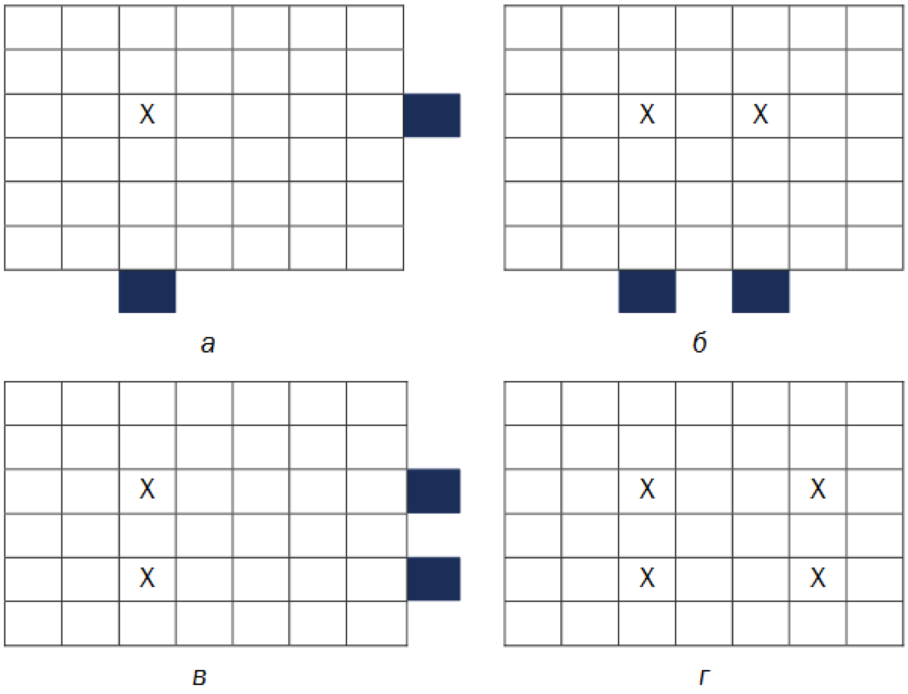

Рис. 5.2. Пояснение к принципу обнаружения местоположения ошибочных битов в принятом сообщении:
а–*г* – варианты местоположения ошибок

Однако этим кодом не могут быть установлены местоположения многократных ошибок, имеющих четное число искаженных символов как по строкам, так и по столбцам (рис. 5.2, *б*, *в*). Простейшая необнаруживаемая ошибка содержит четыре искаженных символа, расположенных в вершинах прямоугольника или квадрата (рис. 5.2, *г*). Это происходит из-за того, что четность (паритет) по строкам и по столбцам матрицы не нарушается. Полезную информацию о кодировании и декодировании информации итеративным кодом можно найти в источниках [10, 11].

5.1.2. Многомерные линейные итеративные коды

Принято считать рассматриваемый код *многомерным*, если количество измерений, по которым вычисляются и анализируются паритеты, не менее 3. Таким образом, простейшим многомерным линейным итеративным кодом является код *трехмерный*. Его достаточно подробное описание и особенности использования можно найти в статьях [12–14].

Пример реализации трехмерного случая иллюстрирует рис. 5.3. Дополнительно к двум кодам на основе кодов простой четности (по вертикали и горизонтали) избыточные символы вычисляются по диагонали: X_d.

Приведенную на рис. 5.3 трехмерную структуру итеративных кодов можно дополнить достаточно большим числом разнообразных проверок на четность по диагоналям, тем самым получив набор кодов с высокими корректирующими возможностями.

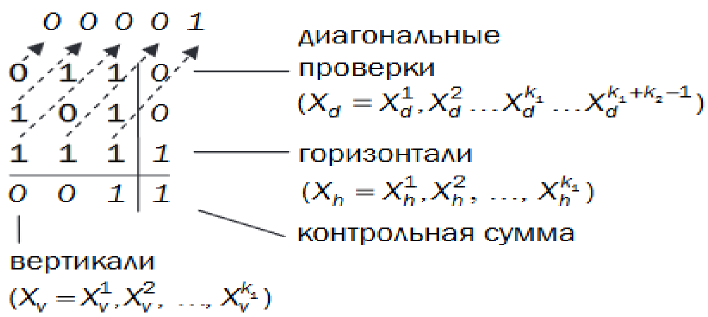

Рис. 5.3. Принцип формирования избыточных символов для линейного итеративного кода с диагональными проверками

Принцип формирования проверочных символов для кодов, построенных на основе вычисления различных диагональных паритетов, при $k = 64$ бита приведен на рис. 5.4 (пять групп линейно независимых паритетов): в первой плоскости, или матрице, записаны первые 16 битов сообщения, во второй плоскости-матрице – символы с 17 по 32 и т. д. Указанные плоскости располагаются по условной оси Z.

Как показано на рис. 5.4, для информационного сообщения из 64 битов ($X_k = x_1, x_2, \ldots, x_{64}$) сформировано 80 битов общего избыточного слова, $X_{rr} = x_{r1}, x_{r2}, \ldots, x_{r80}$:

$$x_{r1} = x_1 + x_2 + x_3 + x_4,$$

$$x_{r2} = x_5 + x_6 + x_7 + x_8, \ldots,$$

$$x_{r5} = x_1 + x_5 + x_9 + x_{13}, \ldots,$$

$$x_{r14} = x_3 + x_8 + x_9 + x_{14}, \ldots,$$

$$x_{r16} = x_1 + x_6 + x_{11} + x_{16},$$

$$x_{r17} = x_{17} + x_{18} + x_{19} + x_{20},\ \ldots,$$
$$x_{r37} = x_{33} + x_{37} + x_{41} + x_{45},\ \ldots,$$
$$x_{r60} = x_{52} + x_{55} + x_{58} + x_{61},\ \ldots,$$
$$x_{r80} = x_{16} + x_{32} + x_{48} + x_{64}.$$

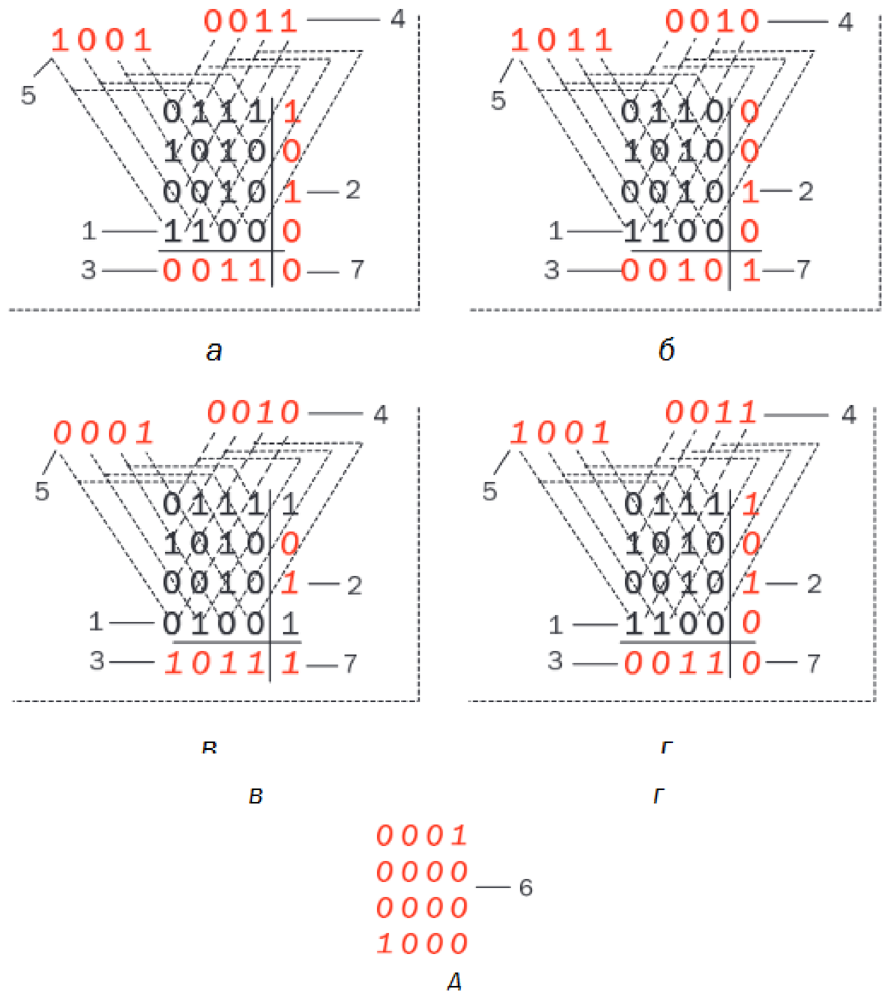

Рис. 5.4. Пояснение к принципу построения кодового слова на основе многомерного линейного итеративного кода с 5 независимыми группами паритетов: *а–г* – плоскости с первой по четвертую соответственно; *д* – паритеты в *z*-плоскости; 1 – информационные биты; 2 – горизонтальные паритеты; 3 – вертикальные паритеты; 4, 5 – соответственно первые и вторые объединенные диагональные паритеты; 6 – *z*-паритеты; 7 – контрольная сумма (X_{hv})

В табл. 5.1 приведены параметры рассмотренного многомерного итеративного кода, которые можно также отнести к классу *низкоплотностных* [15].

Таблица 5.1

Параметры некоторых многомерных итеративных кодов

Длина информационной последовательности (бит), k	Пять линейно независимых паритетов		Семь линейно независимых паритетов		Девять линейно независимых паритетов	
	Избыточность (бит), r	Скорость передачи, $k/(k+r)$	Избыточность (бит), r	Скорость передачи, $k/(k+r)$	Избыточность (бит), r	Скорость передачи, $k/(k+r)$
64	80	0,44	112	0,36	144	0,31
512	320	0,62	448	0,53	576	0,47
4096	1280	0,76	1792	0,70	2304	0,64

5.2. Практическое задание

Разработать собственное приложение, которое позволяет выполнять следующие операции:

1) вписывать произвольное двоичное представление информационного слова X_k (кодируемой информации) длиной k битов в двумерную матрицу размерностью в соответствии с вариантом либо в трехмерную матрицу в соответствии с вариантом (указаны в табл. 5.2);

2) вычислять проверочные биты (биты паритетов): а) по двум; б) по трем; в) по четырем направлениям (группам паритетов);

3) формировать кодовое слово X_n присоединением избыточных символов к информационному слову;

4) генерировать ошибку произвольной кратности (i, $i > 0$), распределенную случайным образом среди символов слова X_n, в результате чего формируется кодовое слово Y_n;

5) определять местоположение ошибочных символов итеративным кодом в слове Y_n в соответствии с используемыми группами паритетов по пункту (2) и исправлять ошибочные символы (результат исправления – слово Y_n');

6) выполнять анализ корректирующей способности используемого кода (количественная оценка) путем сравнения соответствующих слов X_n и Y_n'; результат анализа может быть представлен в виде отношения общего числа сгенерированных кодовых слов с ошибками определенной одинаковой кратности (с одной ошибкой, с двумя ошибками и т. д.) к числу кодовых слов, содержащих

ошибки этой кратности, которые правильно обнаружены и которые правильно скорректированы.

Таблица 5.2

Варианты заданий

Вариант	Длина информационного слова (бит), k	k_1	k_2	z	Количество групп паритетов
1	16	4 8 4 2	4 2 2 4	– – 2 2	2; 3 2; 3 2; 3; 4; 5 2; 3; 4; 5
2	20	4 2 2 2	5 10 5 2	– – 2 5	2; 3 2; 3 2; 3; 4; 5 2; 3; 4; 5
3	24	4 3 3 6	6 8 3 2	– – 4 2	2; 3 2; 3 2; 3; 4; 5 2; 3; 4; 5
4	32	4 2 8 4	8 16 2 4	– – 2 2	2; 3 2; 3 2; 3; 4; 5 2; 3; 4; 5
5	40	5 4 5 2	8 10 4 10	– – 2 2	2; 3 2; 3 2; 3; 4; 5 2; 3; 4; 5

Пример. Для определенного кодового слова X_n с выбранными группами вычисленных избыточных символов сгенерировано N_1 вариантов этого кодового слова (теперь это слово мы обозначаем Y_n) с 3 ошибками ($i = 3$). Среди N_1 ошибочных кодовых слов в N_2 случаях кратность ошибки идентифицирована правильно ($N_2 \leq N_1$) и в N_3 случаях все ошибки скорректированы (слова X_n и Y_n' совпадают). Нужно вычислить соотношения N_2/N_1 и N_3/N_1.

ВОПРОСЫ ДЛЯ КОНТРОЛЯ И САМОКОНТРОЛЯ

1. Охарактеризовать основные параметры итеративного кода.
2. Сравнить основные параметры кодов Хемминга и итеративных кодов.

3. Составить проверочные матрицы кодов Хемминга для кодирования 9-битных сообщений по схеме, представленной на рис. 5.1 (с учетом и без учета символа X_{hv}).

4. Составить проверочные матрицы кодов Хемминга для кодирования 9-битных сообщений по схеме, представленной на рис. 5.3 (с учетом и без учета символа X_{hv}).

5. Составить проверочные матрицы кодов Хемминга для кодирования 9-битных сообщений по схеме, представленной на рис. 5.4 (с учетом и без учета символа X_{hv}).

6. Какое максимальное число ошибок может быть обнаружено итеративным кодом? При каком условии?

7. Определить, какая геометрическая фигура, являющаяся формой для записи символов информационного слова, обеспечивает наименьшую относительную избыточность кодового слова при фиксированном (каком?) k.

8. Результаты оформить в виде отчета по установленным правилам.