УДК 621.391.15:004.7

АЛГОРИТМ ДЕКОДИРОВАНИЯ С ВВОДОМ СТИРАНИЙ ДЛЯ МПП-КОДОВ, ПОСТРОЕННЫХ НАД ПОЛЕМ GF(q)

В. В. Зяблов,
доктор техн. наук, профессор
П. С. Рыбин,
младший научный сотрудник
А. А. Фролов,
младший научный сотрудник
Институт проблем передачи информации им. А. А. Харкевича РАН

Предложен итеративный алгоритм декодирования для кодов с малой плотностью проверок, способный исправлять как ошибки, так и стирания. Представлена зависимость реализуемых корректирующих свойств данного алгоритма от количества стираний. Проведено сравнение данного алгоритма с мажоритарным алгоритмом для случая, когда присутствуют только ошибки.

Ключевые слова *— МПП-код, итеративный алгоритм декодирования, стирание.*

Введение

В настоящее время в связи со все более высокими требованиями к скорости передачи данных особенно интересны кодовые конструкции, для которых существуют быстрые алгоритмы кодирования и декодирования. Естественно, что алгоритмы декодирования должны при этом справляться с большим количеством ошибок.

Двоичные коды с малой плотностью проверок (МПП-коды) на четность были предложены Галлагером [1]. Доказано [2] существование МПП-кодов, способных исправить линейно растущее с длиной кода число ошибок при сложности декодирования $O(n\log_2 n)$, где n — длина кода. В настоящее время эти коды используются в стандартах подвижной беспроводной связи (например, LTE), цифровой телефонии; рекомендованы для использования в стандартах оптической связи, спутниковой связи, WiMAX, 802.11n.

Дальнейшее увеличение скорости передачи возможно лишь только с помощью увеличения «плотности» передаваемой информации (числа бит на герц), так как частотный ресурс ограничен. Одним из способов является увеличение мощности алфавита модуляции. Из-за этого особенно интересными становятся недвоичные корректирующие коды. В работе [3] построены недвоичные МПП-коды и доказан результат, аналогичный результату для двоичных. Описан также мажоритарный алгоритм декодирования для недвоичных МПП-кодов, являющийся обобщением алгоритма «инвертирования бита» для кодов Галлагера. Этот алгоритм способен исправлять только ошибки, однако в некоторых случаях при передаче данных (например, при передаче на многих частотах) в принятом векторе содержатся как ошибки, так и стирания.

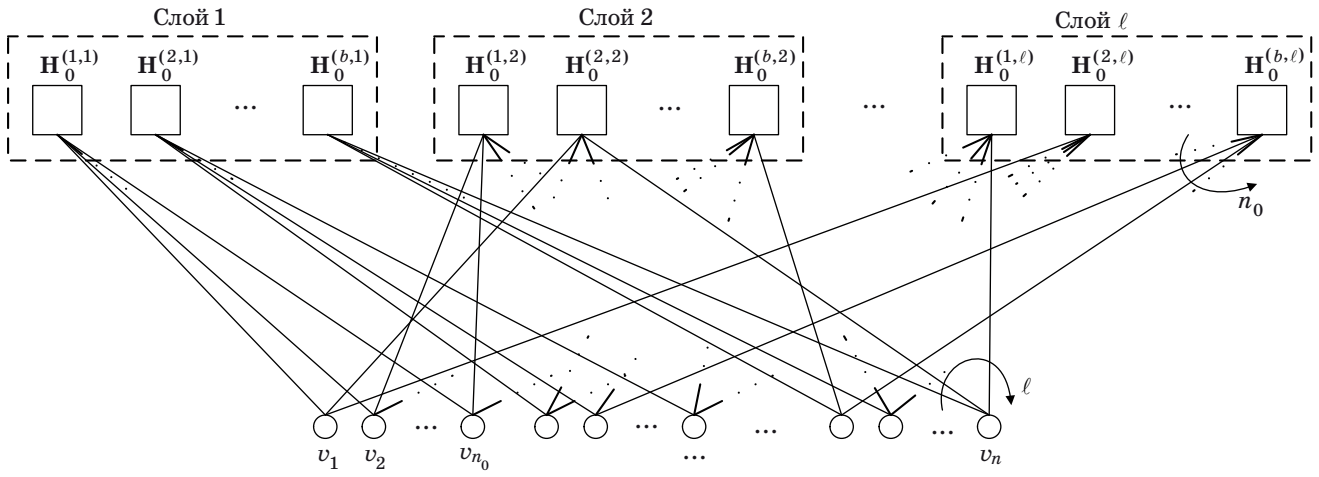
Основной нашей задачей является разработка алгоритма декодирования для МПП-кодов как двоичных, так и недвоичных, способного справляться как с ошибками, так и со стираниями в принятом векторе. Также будут приведены исследования реализуемых корректирующих свойств данного алгоритма, частично представленные в работах [4–8].

Структура МПП-кодов

Для построения проверочной матрицы q-ичного МПП-кода C рассмотрим блочную диагональную матрицу $\mathbf{H}_b$, на главной диагонали которой находятся b проверочных матриц $\mathbf{H}_0$ кода-компонента длины n_0:

$$\mathbf{H}_b = \begin{pmatrix} \mathbf{H}_0 & 0 & \cdots & 0 \\ 0 & \mathbf{H}_0 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & \mathbf{H}_0 \end{pmatrix}_{bm \times bn_0},$$

где m — избыточность кода-компонента ($m = n_0 - k_0$).



■ ***Рис. 1.*** *Граф Таннера для МПП-кода*

Пусть $\phi(\mathbf{H}_b)$ обозначает матрицу, полученную из матрицы $\mathbf{H}_b$ произвольной перестановкой столбцов и умножением их на произвольные ненулевые элементы поля $GF(q)$. Тогда матрица

$$\mathbf{H} = \begin{pmatrix} \phi_1(\mathbf{H}_b) \\ \phi_2(\mathbf{H}_b) \\ \vdots \\ \phi_\ell(\mathbf{H}_b) \end{pmatrix}_{\ell bm \times bn_0}$$

размером $\ell bm \times bn_0$, составленная из ℓ таких матриц, как слоев, является разреженной проверочной матрицей q-ичного МПП-кода.

Замечание 1. Из определения ясно, что длина построенного кода $n = bn_0$.

Замечание 2. Отметим, что каждый символ принятого вектора проверяется в точности ℓ компонентными кодами (ровно одним в каждом слое).

Графически код C можно представить в виде двудольного графа, называемого графом Таннера, в котором символьные вершины соответствуют символам принятого вектора (имеют степень ℓ), а кодовые вершины соответствуют компонентным кодам и имеют степень n_0. Пример такого графа приведен на рис. 1.

Нижняя оценка скорости кода C получена в работе [9]:

$$R \geq 1 - \frac{\ell b(n_0 - k_0)}{bn_0} = 1 - \ell(1 - R_0). \qquad (1)$$

Равенство достигается в случае полного ранга матрицы $\mathbf{H}$. Из соотношения (1) получим ограничение для скорости кода-компонента

$$R_0 > 1 - \frac{1}{\ell},$$

т. е. чем больше количество слоев, тем выше должна быть скорость кода-компонента.

В данной работе будут исследованы МПП-коды с кодом-компонентом, имеющим один проверочный символ. Его проверочная матрица $\mathbf{H}_0$ состоит из ненулевых элементов поля $GF(q)$:

$$\mathbf{H}_0 = \underbrace{(1 \quad \alpha \quad \ldots \quad \alpha^{n_0-1})}_{n_0}, \ \alpha \in GF(q) \setminus \{0\}.$$

Замечание 3. В случае $q = 2$ проверочная матрица кода-компонента имеет вид

$$\mathbf{H}_0 = \underbrace{(1 \quad 1 \quad \ldots \quad 1)}_{n_0}.$$

Алгоритм декодирования с вводом стираний

Главная особенность этого алгоритма состоит во введении стираний на места символов, подозрительных на ошибки. На каждой итерации подозрительные символы заменяются стираниями, и далее в пределах этой итерации выполняется только исправление стираний. Стирания, которые были введены и не были исправлены, после итерации удаляются. Эти операции повторяются до тех пор, пока не случится такого, что в процессе итерации мы не исправили ни одного стирания. В результате выдается либо исправленный вектор, либо отказ от декодирования.

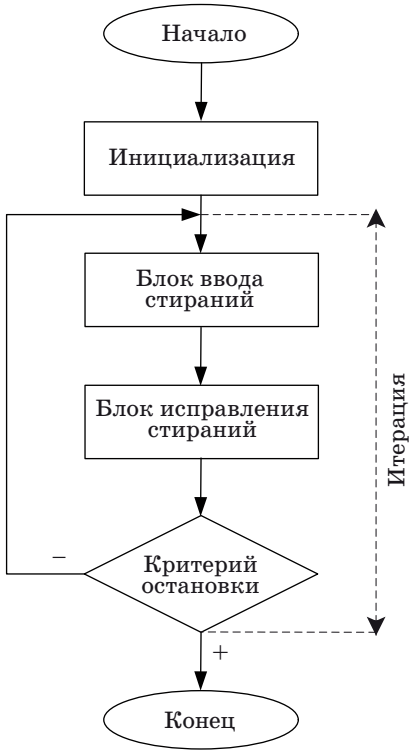
Прежде чем привести формальное описание алгоритма, введем понятие *обобщенного синдрома*. Обобщенный синдром — это вектор, состоящий из синдромов компонентных кодов. Вес обобщенного синдрома — число ненулевых синдромов кодов-компонентов.

Общий случай.

Блок-схема разработанного алгоритма A^* представлена на рис. 2.

Рассмотрим каждый из блоков более подробно.

Инициализация. Вычисляем обобщенный синдром. Он состоит из синдромов компонентных кодов. Если код-компонент содержит стирания,

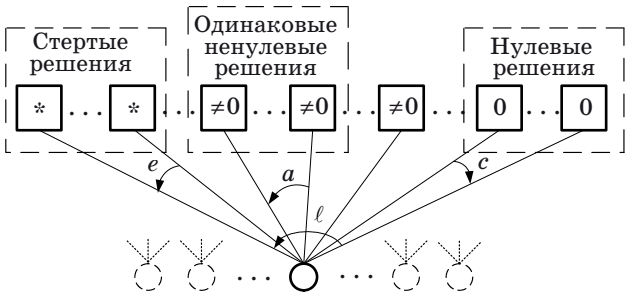


■ ***Рис. 2.*** *Блок-схема разработанного алгоритма декодирования A^**

то его синдром не вычисляется и считается стертым.

Ввод стираний. Для каждого нестертого символа рассматриваются синдромы ℓ кодов-компонентов, в которые входит данный символ. Если синдром кода-компонента ненулевой и нестертый, то вычисляем *решение*. Решением назовем значение, которое нужно добавить к рассматриваемому символу, чтобы синдром кода-компонента стал нулевым. Нулевые и стертые синдромы соответствуют нулевым и стертым решениям (обозначим число нулевых решений через c, число стертых решений — через e). Выбирается подмножество одинаковых ненулевых и нестертых решений максимальной мощности a (если таких подмножеств несколько, то выбирается любое из них). Если $a > c + e$, то на место рассматриваемого символа вводится стирание; синдромы кодов-компонентов, содержащих данный символ, помечаются как стертые; позиция символа добавляется в список стертых символов (рис. 3).

Исправление стираний. Для каждого стертого символа рассматриваются синдромы ℓ кодов-компонентов, в которые входит данный символ. Нас интересуют только коды-компоненты, содержащие ровно одно стирание. Для каждого из таких кодов исправим стирание (заметим, что это очень простая операция), после чего сформируем список возможных значений символа. Выбирается наиболее часто встречающееся значение. Это значение присваивается символу.



■ ***Рис. 3.*** *Введение стираний на места символов, подозрительных на ошибки*

Критерий остановки. Добавленные стирания, которые не были исправлены, удаляются. Сравниваются синдромы до и после итерации. В случае если синдром изменился, перейти к следующей итерации, иначе — вычислить вес синдрома. Если вес нулевой, выдать исправленный вектор, иначе — отказ от декодирования.

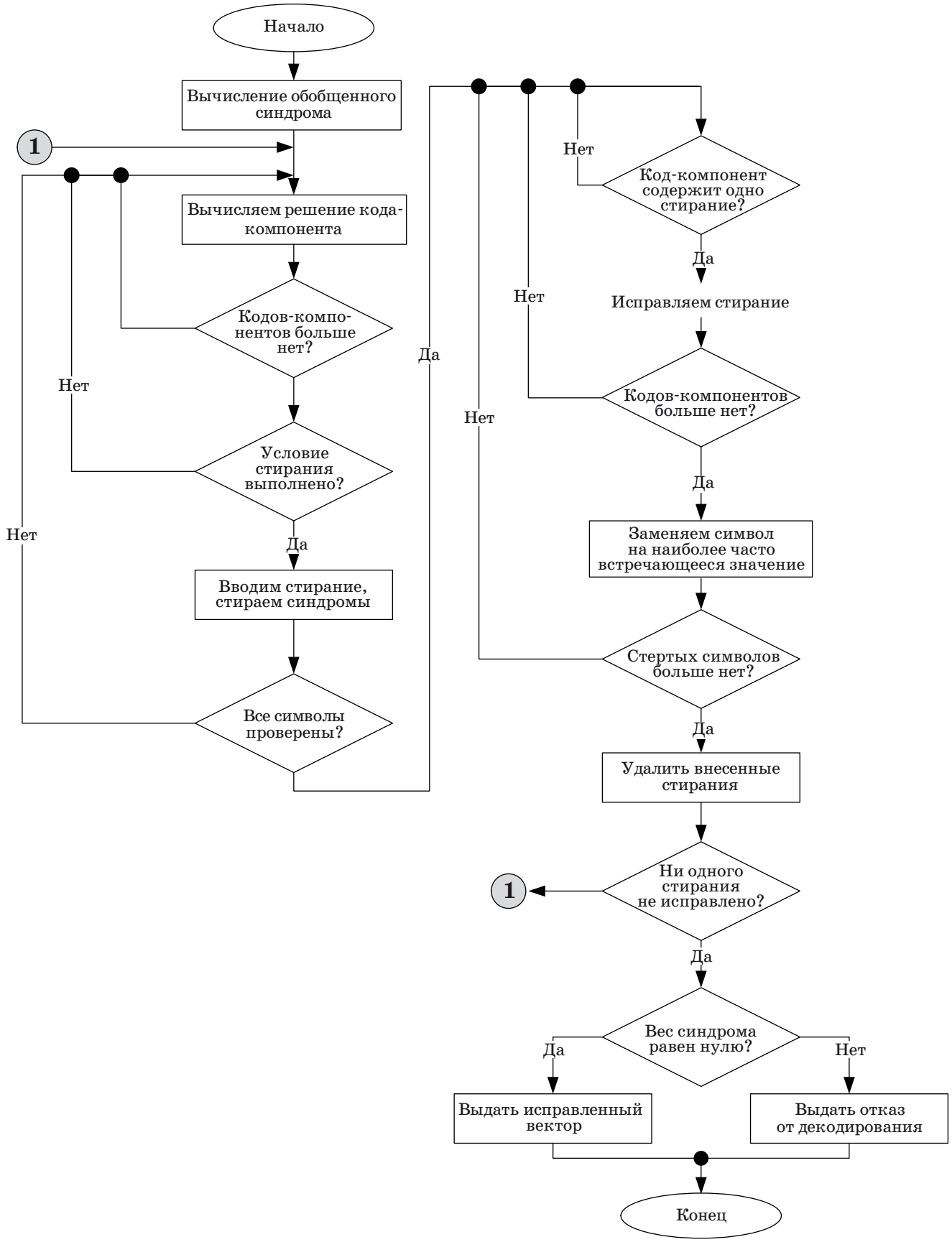
Замечание 4. Этот алгоритм без всяких изменений может быть применен к МПП-кодам с более мощными кодами-компонентами. Однако в случае более мощного кода-компонента можно исправить более чем одно стирание (а именно $d_0 - 1$, где d_0 — кодовое расстояние компонентного кода). В связи с этим разумно изменить алгоритм декодирования (блок исправления стираний).

Замечание 5. В случае $q = 2$ условие $a > c + e$ трансформируется в $a > \ell / 2$. Моделирование показало, что в этом случае вводится много стираний, что приводит к большой вероятности отказа от декодирования.

В следующем разделе описано, как модифицировать алгоритм при $q = 2$, в окончание же этого раздела приведем более подробную блок-схему разработанного алгоритма (рис. 4).

Случай $q = 2$.

Отличие алгоритма A^* для двоичного МПП-кода заключается только в критерии ввода стирания. Как отмечалось выше, условие ввода стирания $a > c + e$ алгоритма A^* трансформируется в двоичном случае в $a > \ell / 2$ (первый критерий). Однако моделирование показало, что в случае большого количества ошибок использование данного критерия приводит к большой вероятности отказа от декодирования. Поэтому был разработан дополнительный (второй) критерий ввода стирания и было решено использовать оба критерия. Критерий ввода стирания изменяется в течение декодирования принятой последовательности. На начальном этапе декодирования принятой последовательности используется первый критерий. Если использование первого критерия ввода стирания привело к отказу от декодирова-



■ ***Рис. 4.*** *Подробная блок-схема алгоритма A**

ния, то происходит замена его на второй критерий. Если же использование второго критерия ввода стирания привело к отказу от декодирования, то происходит выход из цикла с отказом от декодирования. Таким образом, критерий ввода стирания изменяется только один раз в течение декодирования принятой последовательности, а декодирование каждой принятой последовательности начинается с использованием первого критерия ввода стирания.

В этой работе мы рассматривали следующие два критерия ввода стирания для алгоритма A^* декодирования двоичного МПП-кода. Символ заменяется стиранием, если он входит:

1) в более чем $\ell / 2$ невыполненных проверок;

2) в максимальное число невыполненных проверок (если таких символов несколько, то заменяются все).

Остальные шаги алгоритма A^* не требуют модификаций для декодирования двоичного МПП-кода.

Результаты моделирования

Описание моделирования.

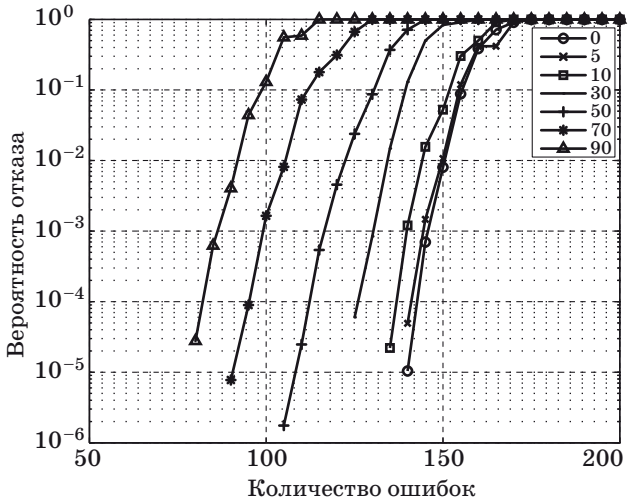
Результат моделирования — зависимость вероятности неправильного декодирования (отказ или переход в другое кодовое слово) от числа ошибок. Сначала задаются параметры кода и начальное число ошибок. Далее генерируется случайный код с заданными параметрами и происходит непосредственно само моделирование, т. е. генерируются случайные векторы ошибок заданного веса и подаются на декодер. Отметим, что мы не производим кодирования, а генерируем только векторы ошибок, т. е. в результате декодирования мы должны получить вектор из всех нулей. После 10 неправильных декодирований число ошибок уменьшается на величину шага. Вероятность неправильного декодирования при заданном числе ошибок вычисляется как отношение числа неправильных декодирований (10 в нашем случае) к общему числу испытаний. Для каждой зависимости было проведено более 10^6 испытаний.

Замечание 6. Отметим, что за все время моделирования не произошло ни одного перехода в другое кодовое слово, т. е. вероятность неправильного декодирования в этом случае равна вероятности отказа.

Результаты для $q = 16$.

Для всех моделирований мы использовали один и тот же код со следующими параметрами: $q = 16$; $n = 2048$; $R = 1/2$; $\ell = 8$. В качестве компонентного кода используется код с $n_0 = 16$.

Сначала посмотрим, как изменяется корректирующая способность исследуемого алгоритма с увеличением начального числа стираний при декодировании с помощью алгоритма A^*. На рис. 5 показано семейство зависимостей, построенных при разном начальном количестве стираний (0, 5, 10, 30, 50, 70, 90). Каждый график представляет собой зависимость вероятности отказа (см. замечание 6) от числа ошибок, число стираний фиксировано.



■ ***Рис. 5.*** *Семейство зависимостей, построенных при разных начальных количествах стираний для $q = 16$*

Введем следующие обозначения:

τ — начальное число стираний;

e^* — число ошибок, при котором вероятность отказа меньше, чем 10^{-4} (выбирается наибольшее число ошибок, удовлетворяющее условию).

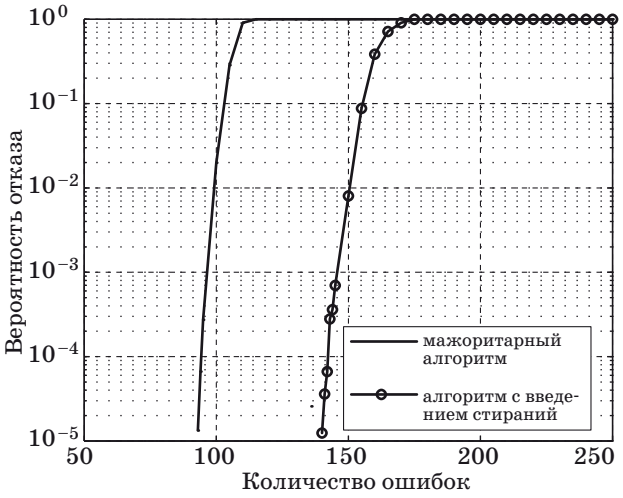
Мы будем использовать величину $d^* = 2e^* + \tau + 1$ для того, чтобы охарактеризовать реализуемую корректирующую способность. Полученная зависимость d^* от начального числа стираний и ее отношение к длине МПП-кода $\delta^* = d^*/n$ приведены в табл. 1.

Как мы видим, величина d^*, характеризующая реализуемую корректирующую способность, уменьшается с увеличением начального числа стираний.

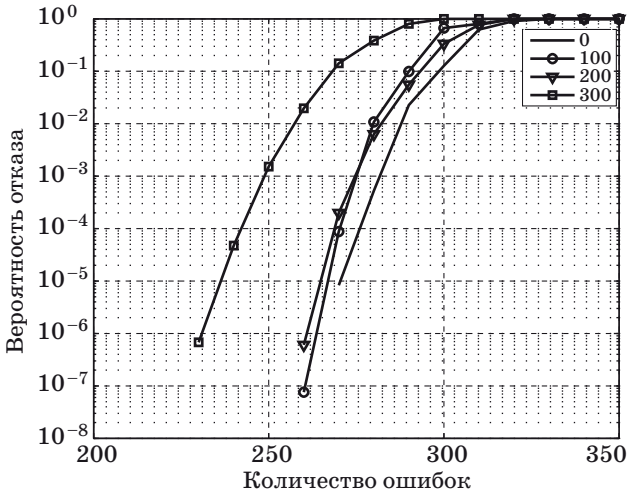
Теперь посмотрим, как алгоритм справляется с ошибками. Пусть начальное число стираний равно нулю, т. е. в принятом векторе есть только

■ ***Таблица 1.*** *Зависимость реализуемой корректирующей способности при $q = 16$ от начального числа стираний*

Показатель	τ						
	0	5	10	30	50	70	90
e^*	142	140	136	126	110	94	81
d^*	285	286	283	283	271	259	253
δ^*	0,139	0,140	0,138	0,138	0,132	0,126	0,124



■ ***Рис. 6.*** *Сравнение результатов моделирования при $q = 16$ для алгоритма с введением стираний и мажоритарного алгоритма*



■ ***Рис. 7.*** *Семейство зависимостей, построенных при разных начальных количествах стираний для $q = 16$*

ошибки. Сравним полученные результаты моделирования и результаты моделирования для мажоритарного алгоритма. Подробное описание мажоритарного алгоритма приведено в работе [3]. Это итеративный алгоритм, на каждой итерации которого решение о замене каждого из просматриваемых символов принимается согласно мажоритарному правилу. Это правило похоже на наш критерий стирания символа и полностью совпадает с ним при отсутствии стертых синдромов кодов-компонентов.

В обоих случаях используется один и тот же МПП-код. Результаты приведены на рис. 6. Как мы видим, полученные результаты для алгоритма с введением стираний оказались лучше.

Результаты для $q = 2$.

Для всех моделирований мы использовали один и тот же код со следующими параметрами: $q = 2$; $n = 7995$; $R = 1/2$; $\ell = 7$. В качестве компонентного кода используется двоичный код с проверкой на четность с $n_0 = 15$.

На рис. 7 приведено семейство зависимостей, построенных при разном начальном количестве стираний (0, 100, 200, 300). Каждый график представляет собой зависимость вероятности отказа от числа ошибок, число стираний фиксировано.

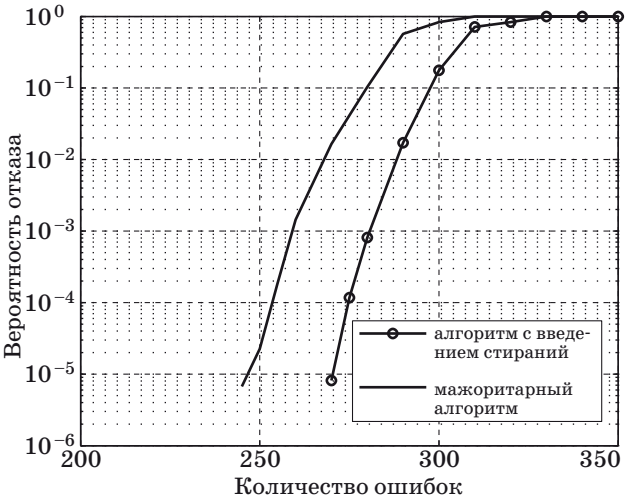
Рассмотрим реализуемую корректирующую способность исследуемого алгоритма, характеризуемую величиной d^*, введенной в предыдущем разделе. Полученные зависимости d^* и δ^* приведены в табл. 2.

Здесь в отличие от результатов предыдущего раздела d^* растет с увеличением начального числа стираний.

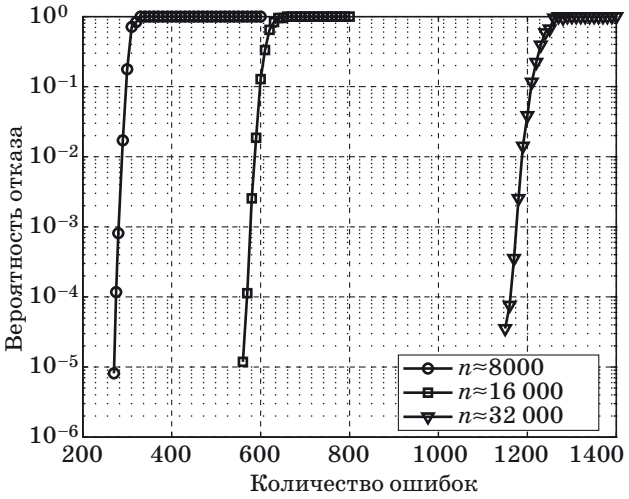
Теперь сравним полученные результаты моделирования и результаты моделирования для мажоритарного алгоритма декодирования двоичных МПП-кодов, подробно описанного в работе [10]. В обоих случаях используется один и тот же МПП-код. Результаты приведены на рис. 8. Как мы видим, для алгоритма с введением стираний они оказались лучше.

■ ***Таблица 2.*** *Зависимость реализуемой корректирующей способности при $q = 2$ от начального числа стираний*

Показатель	τ			
	0	100	200	300
e^*	276	271	269	242
d^*	553	643	739	785
δ^*	0,069	0,080	0,092	0,098



■ ***Рис. 8.*** *Сравнение результатов моделирования при $q = 2$ для алгоритма с введением стираний и мажоритарного алгоритма*



■ ***Рис. 9.*** *Семейство зависимостей, построенных для различных длин двоичного МПП-кода*

Рассмотрим теперь, как изменяется доля исправленных ошибок с заданной вероятностью отказа при увеличении длины МПП-кода. На рис. 9 представлено семейство зависимостей вероятности отказа от количества ошибок для различных длин МПП-кода.

Обозначим через W количество ошибок, при которых вероятность отказа равна 10^{-4}, а через ω — ее отношение к длине МПП-кода n. Зависимость ω от n МПП-кода следующая:

n	8000	16 000	32 000
ω	0,0345	0,0356	0,0363

Как мы видим, доля исправленных ошибок увеличивается с увеличением длины МПП-кода.

Заключение

Предложен итеративный алгоритм декодирования МПП-кодов, способный исправлять как ошибки, так и стирания.

Проведено экспериментальное исследование предложенного алгоритма декодирования. Этот алгоритм дает лучшие по сравнению с мажоритарными алгоритмами результаты для канала, в котором есть только ошибки.

Литература

1. **Галлагер Р. Дж.** Коды с малой плотностью проверок на четность. — М.: Мир, 1966. — 144 с.
2. **Зяблов В. В., Пинскер М. С.** Оценка сложности исправления ошибок низкоплотностными кодами Галлагера // Пробл. передачи информ. 1975. Т. 11. № 1. С. 23–36.
3. **Фролов А. А., Зяблов В. В.** Асимптотическая оценка доли ошибок, исправляемых q-ичными МПП-кодами // Пробл. передачи информ. 2010. Т. 46. № 2. С. 47–65.
4. **Зяблов В. В., Рыбин П. С.** Исправление стираний низкоплотностными кодами Галлагера // ИТиС'08, Геленджик, 29 сентября — 03 октября 2008 г. / ИППИ РАН. М., 2008. С. 167–172.
5. **Zyablov V., Rybin P.** Decoding with Erasure Insertion of Binary LDPC Codes // The XII Symp. Problems of redundancy in information and control systems, St. Petersburg, Russia, 26–30 May 2009 / Saint-Peterburg State University of Aerospace Instrumentation. Saint-Peterburg, 2009. P. 150–154.
6. **Zyablov V., Rybin P.** Majority Decoding and Decoding with Erasure Insertion of Binary LDPC codes // Twelfth Intern. Workshop on Algebraic and Combinatorial Coding Theory (ACCT 2010), Novosibirsk, Russia, 5–11 Sept. 2010 / Sobolev Institute of Mathematics SB RAS. Novosibirsk, 2010. P. 329–334.
7. **Frolov A. A., Zyablov V. V.** The application of q-ary LDPC-codes for fiber optic lines // The XII Symp. Problems of redundancy in information and control systems, St. Petersburg, Russia, 26–30 May, 2009 / Saint-Peterburg State University of Aerospace Instrumentation. Saint-Peterburg, 2009. P. 121–125.
8. **Frolov A. A., Zyablov V. V.** Insertion of Erasures as a Method of Q-ry LDPC Codes Decoding // Twelfth Intern. Workshop on Algebraic and Combinatorial Coding Theory (ACCT 2010), Novosibirsk, Russia, 5–11 Sept. 2010 / Sobolev Institute of Mathematics SB RAS. Novosibirsk, 2010. P. 138–143.
9. **Tanner R. M.** A Recursive Approach to Low Complexity Codes // IEEE Trans. Inform. Theory. 1981. Vol. 27. N 5. P. 533–547.
10. **Sipser M., Spielman D. A.** Expander Codes // IEEE Trans. Inform. Theory. 1996. Vol. 42. N 6. P. 1710–1722.