

ТЕОРЕТИКО-ГРАФОВЫЙ МЕТОД ДЕКОДИРОВАНИЯ НЕКОТОРЫХ ГРУППОВЫХ MLD-КОДОВ

В. М. Деундяк^{1,2,a}, Е. А. Лелюк^{1,b}

¹ Институт математики, механики и компьютерных наук им. И. И. Воровича,
ул. Мильчакова, 8а, 344058 Ростов-на-Дону, Россия

² Научно-исследовательский институт «Спецвузавтоматика»,
Газетный пер., 51, 344002 Ростов-на-Дону, Россия

E-mail: ^avl.deundyak@gmail.com, ^blelukevgeniy@mail.ru

Аннотация. Построен класс мажоритарно-декодируемых групповых кодов с помощью метода комбинирования, основанного на применении тензорного произведения и суммы кодов. Конструкция этого класса базируется на известном подходе Касами — Лина, при котором рассматриваются не отдельно взятые коды, а семейства кодов, и использует важную для мажоритарно-декодируемых кодов конструкцию M -ортогональности, предложенную Мэсси. Исследуемые в работе коды являются идеалами в групповых алгебрах над, вообще говоря, некоммутативными конечными группами. Для рассматриваемых групповых кодов разработана алгоритмическая модель мажоритарного декодирования на основе теоретико-графового подхода. Важной частью этой модели является построение специального декодирующего графа для декодирования одной координаты зашумлённого кодового слова, соответствующей этому графу. Групповые свойства кодов позволяют быстро находить декодирующие графы для остальных координат. Разработан алгоритм декодирования, который, обращаясь к декодирующим графам, исправляет ошибки во всех координатах зашумлённого кодового слова. В качестве примера семейств групповых кодов приводятся важные в криптографии двоичные коды Рида — Маллера. Кодовые криптосистемы рассматриваются как альтернатива широко применяемым в настоящее время теоретико-числовым криптосистемам, поскольку оказываются стойкими к атакам с помощью квантовых компьютеров. Актуальность решаемых в работе задач заключается в том, что использование групповых кодов и их различных комбинаций в настоящее время является одним из перспективных способов укрепления кодовых криптосистем, поскольку позволяет строить новые коды со сложной алгебраической структурой, что

положительно сказывается на стойкости кодовой криптосистемы. Ил. 2, библиогр. 18.

Ключевые слова: MLD-код, мажоритарное декодирование, групповой код, тензорное произведение, граф.

Введение

Среди комбинаторных методов декодирования линейных кодов, таких как декодирование по синдромам, декодирование по информационным совокупностям и мажоритарное декодирование Мэсси, особо выделяется последний метод [1]. Особенность его состоит в том, что сложность декодирования остаётся низкой при уменьшении скорости кода за счёт возможности одновременного вычисления нескольких символов [2, с. 134], в то время как первые два метода применимы для кодов с большой скоростью. Однако для применения мажоритарного декодера необходимо, чтобы линейный код обладал некоторой специальной структурой. К таким кодам относятся введённые Мэсси MLD-коды и, в частности, классические коды Рида — Маллера, которые применяются как в телекоммуникации, так и в криптографии [3]. Для групповых кодов, являющихся идеалами в групповых алгебрах, метод мажоритарного декодирования рассмотрен в монографии [4].

Мажоритарные методы декодирования представляются актуальными с точки зрения их применения в криптографии для построения новых кодовых криптосистем, которые наряду с криптосистемами на решётках набирают популярность в связи с развитием постквантовой криптографии [5]. Отметим однако, что среди недостатков существующих кодовых криптосистем отмечают большую длину ключа. Попытки её уменьшить за счёт применения новых кодов зачастую приводили к тому, что полученные криптосистемы оказывались нестойкими [6–9]. Представляется, что усиление стойкости кодовых криптосистем к структурным атакам при сохранении приемлемой длины ключа возможно путём использования помехоустойчивых кодов, которые не обладают явно выраженной алгебраической структурой. Однако для использования таких кодов в криптосистемах необходимо наличие быстрого алгоритма декодирования. Подобный подход для построения новых криптографических систем применяется в [10, 11], где, в частности, разработаны новые декодеры на деревьях как для MLD-кодов, так и для тензорного произведения MLD-кодов.

В настоящей работе построены групповые MLD-коды при помощи предложенного Касами и Лином метода комбинирования кодов, основанного на применении тензорного произведения и суммы кодов. Эти коды представляют собой перспективную альтернативу классическим

кодам с точки зрения сопротивления к структурным атакам на кодовые криптосистемы. Для построенных кодов разработан мажоритарный метод декодирования, который является обобщением и развитием подходов из [10, 11], в частности, при декодировании используются не деревья, а более общие теоретико-графовые конструкции.

В разд. 1 приводятся основные положения подхода Касами — Лина [12] в удобном для дальнейшего виде, в частности, вводится понятие D -кода. В разд. 2 строятся алгоритмы для декодирования кодов из M -ортогонального семейства с помощью теоретико-графового метода. В разд. 3 приводятся необходимые сведения о групповых кодах, на основе чего в разд. 4 получен основной результат статьи — алгоритмическая модель декодирования групповых D -кодов.

1. Подход Касами — Лина

1.1. l -MLD-коды. Пусть \mathbb{F}_q^n — векторное пространство над полем Галуа \mathbb{F}_q . Зафиксируем в \mathbb{F}_q^n стандартный базис B и связанную с этим базисом метрику Хэмминга. Для $\mathbf{x} \in \mathbb{F}_q^n$ множество базисных векторов, коэффициенты при которых в разложении $\mathbf{x} = \sum_{\mathbf{b} \in B} x_{\mathbf{b}} \mathbf{b}$ ненулевые, называется *носителем вектора \mathbf{x} относительно базиса B* и обозначается $\text{supp}_B(\mathbf{x})$; коэффициент $x_{\mathbf{b}}$ будем называть *значением \mathbf{b} -координаты* вектора \mathbf{x} . Вес $\text{wt}_B(\mathbf{x})$ вектора \mathbf{x} определяется как $|\text{supp}_B(\mathbf{x})|$. (Здесь и далее символом $|A|$ обозначается мощность множества A .) Линейное подпространство C пространства \mathbb{F}_q^n называется *линейным кодом* [13]. Пусть $k(C)$ и $n(C) = n$ — размерность и длина кода соответственно, а $d_B(C)$ — минимальное кодовое расстояние кода C . Тогда код C называют $[n(C), k(C), d_B(C)]$ -кодом. Двойственный к коду C код обозначим для удобства через \overline{C} . Тензорное (прямое) произведение $C_1 \otimes C_2$ двух $[n(C_i), k(C_i), d_{B_i}(C_i)]$ -кодов $C_i \subset \mathbb{F}_q^{n_i}$, где $i \in \{1, 2\}$, а B_i — базис в $\mathbb{F}_q^{n_i}$, является $[n(C_1)n(C_2), k(C_1)k(C_2), d_{B_1}(C_1)d_{B_2}(C_2)]$ -кодом [14, п. 6.2.3].

Множество векторов $\{\mathbf{v}^{(1)}, \dots, \mathbf{v}^{(r)}\} \subset \mathbb{F}_q^n$ называется *M -ортогональным* вектору $\mathbf{v} \in \mathbb{F}_q^n$ и обозначается через $\mathcal{M}_{\mathbf{v}}$ [1], если выполняется неравенство $|\text{supp}_B(\mathbf{v}^{(i)})| > |\text{supp}_B(\mathbf{v})|$ для всех $i = 1, \dots, r$ и

$$\text{supp}_B(\mathbf{v}^{(i)}) \cap \text{supp}_B(\mathbf{v}^{(j)}) = \text{supp}_B(\mathbf{v}) \quad \forall i \neq j. \quad (1)$$

Для $U \subset \mathbb{F}_q^n$ через $L(U)$ будем обозначать линейную оболочку множества U в пространстве \mathbb{F}_q^n . В соответствии с подходом Касами — Лина [12] рассмотрим M -ортогональное семейство линейных кодов

$$\mathcal{S} = \{C(0), C(1), \dots, C(J)\} \quad (2)$$

из пространства \mathbb{F}_q^n , для которого выполняются следующие условия: для любого $j \in \{1, \dots, J\}$ существуют $i \in \{0, \dots, j-1\}$ и $U_{ij} \subset C(i)$ такие, что

$$(s1) \ C(0) = \mathbb{F}_q^n, \ C(J) = \{\bar{0}\},$$

$$(s2) \ C(i) = L(U_{ij}),$$

(s3) $\exists d_{ij} \in \mathbb{N} \ \forall \mathbf{u} \in U_{ij} \subset C(i) \ \exists \mathcal{M}_{\mathbf{u}} = \{\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_{d_{ij}-1}\} \subset C(j)$, при этом $d_{ij} \leq d_B(C(i))$.

В соответствии с [12, с. 600], условие (s3) будем записывать в виде

$$C(i) \xleftarrow{d_{ij}} C(j).$$

Согласно [12] код $\overline{C}(j)$ называется *l-MLD-кодом* (или MLD-кодом), если существует такая последовательность $j_1 = 0, j_2, \dots, j_l = j$, что

$$C(j_k) \xleftarrow{d_{j_k j_{k+1}}} C(j_{k+1}), \quad (3)$$

при этом если $d_B(\overline{C}(j)) = \min(d_{j_k j_{k+1}})$, то мажоритарный декодер полностью реализует минимальное кодовое расстояние кода, т. е. позволяет исправить $\lfloor (d_B(\overline{C}(j)) - 1)/2 \rfloor$ ошибок [2, лемма 1; 12, с. 601]. В разд. 2 понятие *l-MLD-кода* будет интерпретировано с точки зрения декодирования.

1.2. D-конструкция. Рассмотрим два семейства кодов

$$\mathcal{S}_1 = \{C_1(0), C_1(1), \dots, C_1(J_1)\}, \quad \mathcal{S}_2 = \{C_2(0), C_2(1), \dots, C_2(J_2)\}$$

из пространств $\mathbb{F}_q^{n_1}$ и $\mathbb{F}_q^{n_2}$ соответственно, удовлетворяющих условиям (s1)–(s3) и дополнительно условию

$$(s4) \ C_t(0) \supset C_t(1) \supset C_t(2) \supset \dots \supset C_t(J), \ t \in \{1, 2\}.$$

Пусть $D_0 = \{(i, j) \mid i = 0, \dots, J_1, j = 0, \dots, J_2\}$. Для любого $D \subseteq D_0$ определим код длины $n_1 n_2$:

$$C(D) = L\left(\bigcup_{(i,j) \in D} C_1(i) \otimes C_2(j)\right), \quad C_1(i) \in \mathcal{S}_1, \ C_2(j) \in \mathcal{S}_2. \quad (4)$$

Далее будем рассматривать только множества $D \subset D_0$, удовлетворяющие условию: если $(i, j) \in D$, $k \geq i$, $s \geq j$, то $(k, s) \in D$. Набор таких множеств D обозначим через $F(D_0)$. Если $D \in F(D_0)$, то *D-кодом* будем называть код $\overline{C}(D)$, двойственный коду $C(D)$. Задача о декодировании *D-кодов* решается в разд. 4.

По множеству $D \in F(D_0)$ определим следующие множества:

$$D^* = \{(i, j) \mid (i-1, j-1) \notin D\},$$

$$D_b = \{(i, j) \mid (i, j) \in D, (i-1, j) \notin D, (i, j-1) \notin D\},$$

$$D_b^* = \{(i, j) \mid (i, j) \in D^*, (i, j+1) \notin D^*, (i+1, j) \notin D^*\}.$$

Отметим, что $(D_b)^* \neq D_b^*$, $(D_b)^* \neq (D^*)_b$, $(D^*)_b \neq D_b^*$.

Пример 1. Если $D = \{(i, j) \mid i + j > \mu\}$, то $D^* = \{(i, j) \mid i + j \leq \mu + 2\}$, $D_b = \{(i, j) \mid i + j = \mu + 1\}$, $D_b^* = \{(i, j) \mid i + j = \mu + 2\}$ (см. [12, с. 604]).

Пример 2. Если $D = \{(i, j) \mid i \geq J_1 - 1, j \geq J_2 - 1\}$, то $D^* = \{(i, j) \mid i \leq J_1 - 1, j \leq J_2 - 1\}$, $D_b = \{(J_1 - 1, 0), (0, J_2 - 1)\}$, $D_b^* = \{(J_1 - 1, J_2 - 1)\}$ (см. [12, пример 3]).

Следующие леммы доказаны в [12].

Лемма 1. Для произвольного $D \in F(D_0)$

$$k(C(D)) = \sum_{(i,j) \in D} k_1(i)k_2(j),$$

где $k_1(i) = k(C_1(i)) - k(C_1(i+1))$, $k_2(j) = k(C_2(j)) - k(C_2(j+1))$.

Лемма 2. Пусть $d_i^t = d_{B_t}(\overline{C}_t(i))$ — минимальное кодовое расстояние $\overline{C}_t(i)$, $t \in \{1, 2\}$. Тогда минимальное кодовое расстояние d кода $\overline{C}(D)$ равно

$$d = \min_{(i,j) \in D_b} \{d_i^1 d_j^2\}.$$

Следует отметить, что Э. Л. Блохом и В. В. Зябловым [15] и В. А. Зиновьевым [16] введён класс обобщённых каскадных кодов, который по основным параметрам близок к D -кодам, однако определения класса кодов Блоха — Зяблова — Зиновьева и класса D -кодов существенно отличаются.

Лемма 3. Для любого $D \in F(D_0)$ выполняются равенства

$$C(D) = L\left(\bigcup_{(i,j) \in D_b} C_1(i) \otimes C_2(j)\right),$$

$$\overline{C}(D) = L\left(\bigcup_{(i,j) \in D_b^*} \overline{C}_1(i) \otimes \overline{C}_2(j)\right).$$

Применяя лемму 3, в примере 2 имеем $\overline{C}(D) = \overline{C}_1(J_1 - 1) \otimes \overline{C}_2(J_2 - 1)$ — тензорное произведение кодов. Пример кода такого типа фактически рассматривается в статье [11].

Пример 3. Пусть $RM(r, m) \subset \mathbb{F}_2^n$ — двоичный код Рида — Маллера, где $r, m \in \mathbb{N}$, $r \leq m$, $n = 2^m$ [13]. Далее будем пользоваться тем, что $\overline{RM}(r, m) = RM(m - r - 1, m)$. Рассмотрим семейства кодов

$$\mathcal{S}_1 = \{C_1(0) = RM(m_1, m_1), C_1(1) = RM(m_1 - 1, m_1), \dots,$$

$$C_1(m_1) = RM(0, m_1), C_1(m_1 + 1) = \{\bar{0}\}\},$$

$$\mathcal{S}_2 = \{C_2(0) = RM(m_2, m_2), C_2(1) = RM(m_2 - 1, m_2), \dots, \\ C_2(m_2) = RM(0, m_2), C_2(m_2 + 1) = \{\bar{0}\}\}.$$

Эти семейства удовлетворяют условиям (s1)–(s4). Действительно, условия (s1) и (s4) выполняются в силу определения кода Рида — Маллера, а условия (s2) и (s3) фактически проверяются в [17], где строится набор M -ортогональных систем для кодов Рида — Маллера. В [12, пример 4, с. 606] показано, что если $D = \{(i, j) \mid i + j > \mu\}$, то $\overline{C}(D) = RM(\mu, m_1 + m_2)$, где $\mu = [0, \dots, m_1 + m_2 - 1]$.

2. Декодирование M -ортогонального семейства кодов

В этом разделе определяется конструкция декодирующего графа для M -ортогонального семейства кодов и строятся алгоритмы декодирования по этому графу. Похожая конструкция была разработана в [17] и применялась в [11], где использовались декодирующие деревья. Новая конструкция декодирующих графов, с одной стороны, более удобная для построения алгоритмов, а с другой, более общая и будет использована в разд. 4 для декодирования групповых D -кодов.

Ниже изложен процесс мажоритарного декодирования кода $\overline{C}(J - 1)$, двойственного коду $C(J - 1)$ из семейства \mathcal{S} (см. (2)). Для заданных вектора $\mathbf{b} \in C(0)$ веса 1 и кода $\overline{C}(J - 1)$ определим декодирующий граф $G(\mathcal{S}, \mathbf{b}) = (V, E)$ как связный ориентированный ациклический граф, удовлетворяющий следующим условиям, при формулировке которых используются обозначения из (s1)–(s3).

(g1) В графе существует единственный исток, который помечен парой $(\mathbf{b}, (0, k))$, где $k \in \{1, \dots, J\}$. Остальные вершины графа $v \in V$ помечены парами $(\mathbf{x}, (i, j))$, где $\mathbf{x} \in U_{ij}$, U_{ij} — порождающее множество кода $C(i)$, удовлетворяющее условию (s3).

(g2) Вершины без исходящих дуг помечены парой $(\mathbf{x}, (J - 1, J))$, где $\mathbf{x} \in C(J - 1)$; эти вершины являются стоками графа.

(g3) Любая помеченная вершина $v_1(\mathbf{x}_1, (i, j))$, не являющаяся стоком графа, является началом $d_{ij} - 1$ ориентированных дуг

$$e = (v_1(\mathbf{x}_1, (i, j)), v_2(\mathbf{x}_2, (j, k))) \in E$$

для некоторого фиксированного $k \in \{j + 1, \dots, J\}$, где \mathbf{x}_2 пробегает множество $\mathcal{M}_{\mathbf{x}_1}$ из условия (s3); т. е. вектор \mathbf{x}_2 , которым помечена вершина v_2 , входит в M -ортогональное множество для вектора \mathbf{x}_1 , которым помечена вершина v_1 .

(g4) Из любой вершины существует путь к одному из стоков графа.

Легко убедиться, что существование такого графа является следствием условий (s1)–(s3), однако этот граф в общем случае определяется неоднозначно.

Опишем принцип построения декодирующего графа G для вектора $\mathbf{b} \in C(0)$ веса 1 и кода $\overline{C}(J-1)$ из семейства S при выполнении условий (s1)–(s3) для этого семейства. Декодирующий граф строится итерационно. Сначала для кода $C(0)$ определяем код $C(i)$ из семейства, который удовлетворяет условию $C(0) \xleftarrow{d_{0i}} C(i)$. Затем в коде $C(i)$ выберем множество векторов $M_b = \{u \mid u \in C(i)\}$, $|M_b| = d_{0i} - 1$, M -ортогональное для вектора $\mathbf{b} \in C(0)$. Соединим вершины, соответствующие этим векторам, с помощью дуг следующим образом. Вершина v_b , помеченная вектором b , соединяется с каждой вершиной v_u , помеченной вектором из M -ортогонального множества, таким образом, что вершина v_b является началом дуги, а каждая вершина v_u ($u \in M_b$) является концом соответствующей дуги.

На следующей итерации построения графа для кода $C(i)$ определяем код $C(j)$ из семейства, который удовлетворяет условию $C(i) \xleftarrow{d_{ij}} C(j)$, и для каждого вектора $u \in M_b$ и соответствующей вершины v_u выполняем аналогичные действия: находим M -ортогональные множества для каждого вектора u и соединяем соответствующие вершины дугами, причём мощности этих M -ортогональных множеств будут совпадать. На каждой последующей итерации алгоритма вершины, помеченные векторами из M -ортогональных множеств, являются стоками промежуточного графа. Алгоритм заканчивает своё выполнение, когда стоками графа становятся вершины, помеченные векторами из кода $C(J-1)$, т. е. когда для какого-то промежуточного кода $C(k)$ выполняется условие $C(k) \xleftarrow{d_{k,J-1}} C(J-1)$, а также все вершины, помеченные необходимыми векторами из $C(k)$, соединяются с вершинами, помеченными векторами из $C(J-1)$, составляющими M -ортогональное множество для каждого такого вектора из $C(k)$. Таким образом, направление дуг в результирующем графе определяется описанным выше процессом, при этом вершина, помеченная вектором b , является истоком графа, а вершины, помеченные векторами из кода $C(J-1)$, являются стоками графа, что соответствует условию (g3). Из описанного алгоритма построения графа с учётом направления дуг вытекает выполнение условия (g4).

Отметим, что если все векторы из M -ортогональных множеств, построенных на одной итерации алгоритма, не пересекаются для каждой такой итерации, то декодирующий граф представляет собой дерево. В этом случае на каждой итерации алгоритма достраивается соответствующий уровень дерева. Подробный алгоритм построения декодирующего дерева в частном случае кодов Рида — Маллера описан в [17]. Количество вершин $\kappa(T)$ в дереве T для произвольного кода определяется формулой

где l_k — количество потомков для какого-либо узла дерева на уровне k , поскольку количество потомков для узлов, находящихся на одном уровне, совпадает, а L — высота дерева. Действительно, на нулевом уровне находится ровно одна вершина v_b , являющаяся корнем, количество вершин на первом уровне равно мощности M -ортогонального множества для вектора b , количество вершин на втором уровне дерева равно произведению мощности M -ортогонального множества для вектора b и мощности M -ортогонального множества для одного из векторов, который принадлежит M -ортогональному множеству для вектора b , и т. д.

The diagram illustrates the evolution of a 7-bit binary string over 10 generations. The nodes are arranged in levels, with the bottom level representing the initial state and the top level representing the final state. The nodes are connected by directed edges (arrows) indicating the flow of information or the evolution of the string.

The nodes are labeled as follows:

- Level 0 (Bottom): 10000000, 11000000, 11110000
- Level 1: 10100000, 10010000
- Level 2: 10001000, 10000100, 10000010, 10000001
- Level 3 (Top): 10000001

The connections between levels are as follows:

- Level 0 to Level 1:
 - 10000000 → 10100000
 - 11000000 → 10100000
 - 11110000 → 10100000
 - 10000000 → 10010000
 - 11000000 → 10010000
 - 11110000 → 10010000
- Level 1 to Level 2:
 - 10100000 → 10001000
 - 10100000 → 10000100
 - 10100000 → 10000010
 - 10100000 → 10000001
 - 10010000 → 10001000
 - 10010000 → 10000100
 - 10010000 → 10000010
 - 10010000 → 10000001
- Level 2 to Level 3:
 - 10001000 → 10000001
 - 10000100 → 10000001
 - 10000010 → 10000001
 - 10000001 → 10000001

Рис. 1. Декодирующий граф для кода $RM(1, 3)$

$\overline{C}(J-1)$ может быть меньше, чем в случае дерева, но, как и в случае дерева, определяется мощностью M -ортогональных множеств для векторов, которыми помечены вершины графа. Поскольку код $\overline{C}(J-1)$ является l -MLD-кодом для некоторого $l \leq J-1$, существует такая последовательность $j_1 = 0, j_2, \dots, j_l = J-1$, что выполняется условие $C(j_k) \xleftarrow{d_{j_k j_{k+1}}} C(j_{k+1})$, и, таким образом, количество вершин оценивается по формуле

$$\kappa(G) \leq 1 + \sum_{i=1}^{l-1} \prod_{k=1}^i (d_{j_k j_{k+1}} - 1),$$

где $d_{j_k j_{k+1}} - 1$ — количество потомков, определяемых (3) и условием (s3).

Пример 4. Пусть $S = \{\mathbb{F}_2^8, RM(2, 3), RM(1, 3), \{\bar{0}\}\}$ — семейство кодов Рида — Маллера. Декодировующий граф для вектора $\mathbf{b} = (10000000)$ и кода $\overline{RM}(1, 3) = RM(1, 3)$ изображён на рис. 1.

Ниже с помощью графа $G(\mathcal{S}, \mathbf{b})$ опишем процесс мажоритарного декодирования для кодов семейства \mathcal{S} . Сначала рассмотрим кодовый вектор $\mathbf{c} \in \overline{C}(J-1)$, принятый из канала вектор $\mathbf{x} = \mathbf{c} + \mathbf{e}$, где $\text{wt}_B(\mathbf{e}) \leq \lfloor (d_B(\overline{C}(J-1)) - 1)/2 \rfloor$, $d_B(\overline{C}(J-1))$ — минимальное кодовое расстояние кода $\overline{C}(J-1)$, и рассмотрим разложение $\sum_{\mathbf{b} \in B} e_{\mathbf{b}} \mathbf{b}$ вектора ошибок \mathbf{e} по базису B . Если для \mathbf{b} -координаты построен декодирующий граф $G(\mathcal{S}, \mathbf{b}) = (V, E)$, то значение $e_{\mathbf{b}}$ для \mathbf{b} -координаты вектора \mathbf{e} , как показано ниже, вычисляется однозначно с помощью метода мажоритарного декодирования. Если для каждой координаты кода C существует декодирующий граф, то существует последовательность $j_1 = 0, j_2, \dots, j_l = j$ такая, что выполняется условие $C(j_k) \xleftarrow{d_{j_k j_{k+1}}} C(j_{k+1})$, поэтому этот код является l -MLD-кодом.

Процесс декодирования кода $\overline{C}(J-1)$ в ненулевой координате вектора \mathbf{b} заключается в сопоставлении каждой вершине графа $G = G(\mathcal{S}, \mathbf{b})$ элемента из \mathbb{F}_q , которым будет помечаться соответствующая вершина. Вычисление этих дополнительных «декодировующих» меток выполняется посредством обхода графа, к примеру, с помощью рекурсивного алгоритма. Обход графа начинается с истока. Для каждой рассматриваемой вершины по графу находится множество вершин, являющихся прямыми потомками для данной вершины. Далее в зависимости от того, является полученное множество пустым или нет, выполняются следующие действия. Если множество потомков пусто, то рассматриваемая вершина является стоком и помечена парой $(J-1, J)$ и, соответственно, вектором из кода $C(J-1)$. Для таких вершин вычисляется скалярное произведение принятого по каналу вектора и вектора, которым помечена данная

вершина. Результатом в этом случае будет являться сумма координат вектора ошибок, соответствующих ненулевым координатам вектора, которым помечена данная вершина.

Алгоритм 1 (MajorVote)

Вход: \mathcal{A} — последовательность чисел из \mathbb{F}_q

Выход: элемент $v \in \mathbb{F}_q$, который в последовательности \mathcal{A} встречается наибольшее число раз

В другом случае, когда множество потомков непусто, вычисляется множество «декодирующих» меток потомков M_l . Для каждого потомка выполняется следующая проверка. Если метка потомка вычислена, т. е. алгоритм обхода уже проходил эту вершину, то метка добавляется к множеству M_l . В противном случае для этой вершины выполняются все вышеописанные действия и обход продолжается уже в этой вершине. После вычисления меток всех потомков полученное множество M_l подаётся на вход алгоритму MajorVote из [17], который выполняет голосование. Для полноты изложения этот алгоритм приводится и в данной работе.

Алгоритм 2 (decode_node)

Вход: \mathbf{y} — принятый по каналу зашумлённый кодовый вектор, $v_k(\mathbf{x}_k, (i, j))$ — вершина графа G , G — декодирующий граф для кода $\overline{C}(J-1)$

Выход: $l(v_k) \in \mathbb{F}_q$ — значение метки для вершины v_k

```

1:  $D := \text{descendants}(v_k(\mathbf{x}_k, (i, j)), G)$ 
2: if  $D = \{\emptyset\}$  then
3:    $l(v_k) := (\mathbf{y}, \mathbf{x}_k)$   $\triangleright \mathbf{x}_k$  — вектор, которым помечена вершина  $v_k$ 
4: else
5:    $M_l := \emptyset$ 
6:   for all  $v_l \in D$  do
7:     if  $l(v_l) \neq \emptyset$  then
8:       к  $M_l$  добавить  $l(v_l)$ 
9:     else
10:      к  $M_l$  добавить  $\text{decode\_node}(\mathbf{y}, v_l, G)$ 
11:     end if
12:   end for
13:    $l(v_k) := \text{MajorVote}(M_l)$ 
14: end if
15: пометить вершину  $v_k$  меткой  $l(v_k)$ 
16: return  $l(v_k)$ 

```

Результат голосования является элементом из \mathbb{F}_q , которым и помечается текущая вершина. После обхода всего графа значением «декодирующей» метки истока будет являться значение координаты вектора ошибки, соответствующей вектору \mathbf{b} .

Алгоритм 3 (decode)

Вход: \mathbf{x} — зашумлённый кодовый вектор, G — декодирующий граф для кода $\overline{C}(J-1)$
Выход: $\tilde{c} \in \mathbb{F}_q^n$ — зашумлённый кодовый вектор с декодированной координатой, соответствующей вектору, которым помечен исток $v_0(\mathbf{b}, (0, j))$ графа G
1: $t := \text{decode_node}(\mathbf{x}, v_0(\mathbf{b}, (0, j)), G)$
2: $\tilde{c} = \mathbf{x} + \mathbf{b}_t$ $\triangleright \mathbf{b}_t$ — вектор \mathbf{b} , в котором ненулевая координата заменена на t
3: **return** \tilde{x}

В соответствии с вышеописанным процессом декодирования рассмотрим формальные алгоритмы `decode_node` и `decode` для декодирования координаты зашумлённого кодового слова $\mathbf{x} = \mathbf{c} + \mathbf{e}$, соответствующей вектору $\mathbf{b} \in \mathbb{F}_q^n$, где $\mathbf{c} \in \overline{C}(J-1)$, $\mathbf{e} \in \mathbb{F}_q^n$. Будем использовать алгоритм `descendants(v, G)`, который, получая на вход вершину $v \in G$ и граф G , возвращает множество вершин графа, являющихся прямыми потомками для вершины в графе G . Этот алгоритм ввиду его простоты приводить не будем. К алгоритму `descendants` обращается рекурсивный алгоритм `decode_node`, который выполняет обход графа, пометая его вершины дополнительными «декодирующими» метками. В алгоритме `decode` на вход подаются зашумлённое кодовое слово \mathbf{x} и граф G и осуществляется вызов алгоритма `decode_node` для истока графа. На выходе алгоритма `decode` получается принятый по каналу кодовый вектор, в котором исправлена координата, соответствующая вектору \mathbf{b} .

Пример 5. Проиллюстрируем процесс мажоритарного декодирования на примере кода $RM(1, 3)$ и декодирующего графа $G = (V, E)$, изображённого на рис. 1. Пусть $\mathbf{c} = (01010101) \in \overline{RM}(1, 3) = RM(1, 3)$ — кодовый вектор, $\mathbf{x} = (10010101)$ — принятый по каналу вектор, $\mathbf{e} = (11000000)$ — вектор ошибок. Выполним декодирование первой координаты вектора \mathbf{x} . Для вершин, являющихся стоками графа, вычислим скалярное произведение \mathbf{x} с векторами, которыми помечены эти вершины, и пометим рассмотренные вершины значением скалярного произведения. Для остальных вершин значение дополнительных декодирующих

меток вычисляется с помощью голосования по меткам потомков. Результирующий помеченный граф изображён на рис. 2. Метка истока графа соответствует значению ошибки в первой координате.

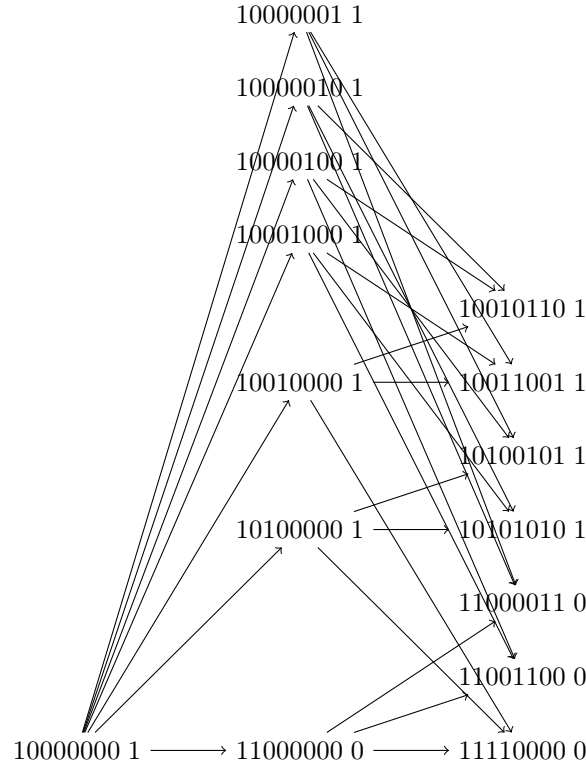


Рис. 2. Помеченный декодирующий граф для кода $RM(1,3)$

3. Групповые коды

Пусть $\mathcal{G} = \{g_1 = \hat{1}, \dots, g_{|\mathcal{G}|}\}$ — конечная группа с зафиксированным линейным порядком $\text{Ord}(\mathcal{G})$ на множестве её элементов, $\hat{1}$ — нейтральный элемент группы. Рассмотрим [4, 18] групповую алгебру $\mathbb{F}_q\mathcal{G}$, элементами которой являются формальные суммы

$$\phi = \sum_{g \in \mathcal{G}} a_g g, \quad a_g \in \mathbb{F}_q. \quad (5)$$

Иногда такие формальные суммы будем называть векторами и использовать обозначение $\phi(g) = a_g$. В конечномерной групповой алгебре $\mathbb{F}_q\mathcal{G}$ зафиксируем базис $B = B^{\mathbb{F}_q\mathcal{G}} := \{\mathbf{g} = \delta_g\}_{g \in \mathcal{G}}$, где $\delta_g = 1g$ — функция

Дирака, $\mathbf{1} := 1\hat{1}$. Это позволяет рассматривать в $\mathbb{F}_q\mathcal{G}$ метрику Хэмминга. Отметим, что в категории конечномерных линейных пространств алгебра $\mathbb{F}_q\mathcal{G}$ естественно изоморфна пространству $\mathbb{F}_q^{|\mathcal{G}|}$, и соответствующий изоморфизм обозначим через $\nu_{\mathcal{G}}$. Отметим также, что $\delta_x\delta_y = \delta_{xy}$, поэтому элементы групповой алгебры можно записывать в виде

$$\sum_{g \in \mathcal{G}} a_g \delta_g, \quad a_g \in \mathbb{F}_q.$$

В соответствии с [4, с. 39] всякий отличный от нулевого левый идеал C в групповой алгебре $\mathbb{F}_q\mathcal{G}$ называется *групповым $\mathbb{F}_q\mathcal{G}$ -кодом* длины $n(C) = |\mathcal{G}|$. Идеал в групповой алгебре $\mathbb{F}_q\mathcal{G}$ является подпространством пространства функций $\mathbb{F}_q\mathcal{G}$, а размерность $k(C)$ кода C — размерность этого подпространства. Пусть $B^C = \{\epsilon_1, \dots, \epsilon_{k(C)}\} \subseteq \mathbb{F}_q\mathcal{G}$ — линейный базис идеала C . Заметим, что порядок $\text{Ord}(\mathcal{G})$ индуцирует порядок на базисе $B^{\mathbb{F}_q\mathcal{G}}$ групповой алгебры $\mathbb{F}_q\mathcal{G}$, что позволяет по строкам выписать порождающую матрицу группового кода C :

$$G_C = \begin{pmatrix} \nu_{\mathcal{G}}(\epsilon_1) \\ \dots \\ \nu_{\mathcal{G}}(\epsilon_{k(C)}) \end{pmatrix}. \quad (6)$$

Декодирования групповых MLD-кодов упрощается за счёт дополнительных алгебраических свойств. Группа \mathcal{G} действует слева на групповой алгебре $\mathbb{F}_q\mathcal{G}$ следующим естественным образом (см. [4, с. 32]):

$$\mathcal{G} \times \mathbb{F}_q\mathcal{G} \ni \left(g, \phi = \sum_{h \in \mathcal{G}} \phi_h h\right) \mapsto \phi g^{-1} := \sum_{h \in \mathcal{G}} \phi_{hg^{-1}} h \in \mathbb{F}_q\mathcal{G}. \quad (7)$$

Отметим, что \overline{C} — также групповой код [4], т. е. левый идеал. В силу этого действие группы \mathcal{G} на $\mathbb{F}_q\mathcal{G}$ по правилу (7) индуцирует действие этой группы на коде \overline{C} . С другой стороны, группа \mathcal{G} действует транзитивно на элементах из $B^{\mathbb{F}_q\mathcal{G}}$ и не нарушает M -ортогональности (см. [17, п. 1.2]). Это позволяет в случае групповых MLD-кодов построить набор декодирующих графов для всех координат только по одному такому графу (соответствующие алгоритмы построены в разд. 4). В частности, если для базисной функции $\mathbf{1} = \delta_{\hat{1}} = 1\hat{1}$ удалось построить декодирующий граф $G(\mathcal{S}, \mathbf{1})$, то граф $G(\mathcal{S}, \mathbf{g})$ для базисной функции $\mathbf{g} = \delta_g = 1g$ может быть построен путём действия элементов $g^{-1} \in \mathcal{G}$ на метки вершин графа $G(\mathcal{S}, \mathbf{1})$ по правилу (7).

Далее будем использовать технику тензорных произведений кодов. Пусть $\mathcal{G} = \{g_1, \dots, g_{|\mathcal{G}|}\}$, $\mathcal{H} = \{h_1, \dots, h_{|\mathcal{H}|}\}$ — конечные группы с зафиксированными на них линейными порядками $\text{Ord}(\mathcal{G})$ и $\text{Ord}(\mathcal{H})$. В групповых алгебрах $\mathbb{F}_q\mathcal{G}$ и $\mathbb{F}_q\mathcal{H}$ зафиксируем базисы $B^{\mathbb{F}_q\mathcal{G}} = \{\delta_{g_1}, \dots, \delta_{g_{|\mathcal{G}|}}\}$

и $B^{\mathbb{F}_q \mathcal{H}} = \{\delta_{h_1}, \dots, \delta_{h_{|\mathcal{H}|}}\}$ соответственно. Рассмотрим тензорное произведение $\mathbb{F}_q \mathcal{G} \otimes_{\mathbb{F}_q} \mathbb{F}_q \mathcal{H}$ групповых алгебр $\mathbb{F}_q \mathcal{G}$ и $\mathbb{F}_q \mathcal{H}$ над полем \mathbb{F}_q (см. [18, с. 79]). Отметим, что $\mathbb{F}_q \mathcal{G} \otimes_{\mathbb{F}_q} \mathbb{F}_q \mathcal{H} = \mathbb{F}_q(\mathcal{G} \times \mathcal{H})$.

В групповых алгебрах $\mathbb{F}_q \mathcal{G}$ и $\mathbb{F}_q \mathcal{H}$ выберем групповые коды $C_1 \subseteq \mathbb{F}_q \mathcal{G}$ и $C_2 \subseteq \mathbb{F}_q \mathcal{H}$ с соответствующими базисами $B^{C_1} = \{\epsilon_1, \dots, \epsilon_{k(C_1)}\}$ и $B^{C_2} = \{\phi_1, \dots, \phi_{k(C_2)}\}$, где

$$\epsilon_i = \sum_{g \in \mathcal{G}} a_{i,g} \delta_g, \quad a_{i,g} \in \mathbb{F}_q, \quad \phi_j = \sum_{h \in \mathcal{H}} b_{j,h} \delta_h, \quad b_{j,h} \in \mathbb{F}_q.$$

Тензорным произведением кодов C_1 и C_2 будем называть код $C_1 \otimes C_2 = C_1 \otimes_{\mathbb{F}_q} C_2 \subseteq \mathbb{F}_q(\mathcal{G} \times \mathcal{H})$ с базисом $B^{C_1 \otimes C_2} = \{\epsilon_i \otimes \phi_j \mid i = 1, \dots, k(C_1), j = 1, \dots, k(C_2)\}$, где $(\epsilon_i \otimes \phi_j)(g, h) = \epsilon_i(g) \phi_j(h)$, $(g, h) \in \mathcal{G} \times \mathcal{H}$. Пусть

$$A \otimes B = \begin{pmatrix} a_{1,1}B & \dots & a_{1,s}B \\ a_{2,1}B & \dots & a_{2,s}B \\ \dots & \dots & \dots \\ a_{r,1}B & \dots & a_{r,s}B \end{pmatrix}$$

— тензорное произведение $(r \times s)$ -матрицы $A = (a_{i,j})$ и матрицы B . Тогда $G_{C_1 \otimes C_2} = G_{C_1} \otimes G_{C_2}$, где согласно (6)

$$G_{C_1} = \begin{pmatrix} a_{1,g_1} & \dots & a_{1,g_{|\mathcal{G}|}} \\ a_{2,g_1} & \dots & a_{2,g_{|\mathcal{G}|}} \\ \dots & \dots & \dots \\ a_{k(C_1),g_1} & \dots & a_{k(C_1),g_{|\mathcal{G}|}} \end{pmatrix},$$

$$G_{C_2} = \begin{pmatrix} b_{1,h_1} & \dots & b_{1,h_{|\mathcal{H}|}} \\ b_{2,h_1} & \dots & b_{2,h_{|\mathcal{H}|}} \\ \dots & \dots & \dots \\ b_{k(C_2),h_1} & \dots & b_{k(C_2),h_{|\mathcal{H}|}} \end{pmatrix}.$$

В силу [14, п. 6.2.3] $k(C_1 \otimes C_2) = k(C_1)k(C_2)$, $n(C_1 \otimes C_2) = n(C_1)n(C_2)$,

$$d_{B^{\mathbb{F}_q(\mathcal{G} \times \mathcal{H})}}(C_1 \otimes C_2) = d_{B^{\mathbb{F}_q \mathcal{G}}}(C_1) d_{B^{\mathbb{F}_q \mathcal{H}}}(C_2), \quad (8)$$

где

$$B^{\mathbb{F}_q(\mathcal{G} \times \mathcal{H})} = \{\delta_{(g_1, h_1)}, \dots, \delta_{(g_1, h_{|\mathcal{H}|})}, \delta_{(g_2, h_1)}, \dots, \delta_{(g_{|\mathcal{G}|}, h_{|\mathcal{H}|})}\} \quad (9)$$

— базис групповой алгебры $\mathbb{F}_q(\mathcal{G} \times \mathcal{H})$ на группе $\mathcal{G} \times \mathcal{H}$ с линейным порядком, индуцированным линейными порядками $\text{Ord}(\mathcal{G})$ и $\text{Ord}(\mathcal{H})$.

4. Теоретико-графовый метод декодирования групповых D -кодов

4.1. Групповые D -коды. В качестве M -ортогонального семейства кодов, описанного в (2), можно рассматривать семейство групповых кодов. Пусть $\mathcal{G} = \{g_1 = \hat{1}, \dots, g_{|\mathcal{G}|}\}$ — конечная группа с зафиксированным

линейным порядком на множестве её элементов. Рассмотрим групповую алгебру $\mathbb{F}_q\mathcal{G}$. Тогда семейство $\mathcal{S} = \{C(0), C(1), \dots, C(J)\}$, $C(i) \in \mathbb{F}_q\mathcal{G}$, удовлетворяющее условиям (s1)–(s4), представляет собой M -ортогональное семейство групповых кодов. Примером такого семейства является семейство кодов Рида — Маллера — Бермана [17]. Как отмечалось в разд. 3, поскольку групповой код — левый идеал, действие группы \mathcal{G} на $\mathbb{F}_q\mathcal{G}$ по правилу (7) индуцирует действие этой группы на групповом коде. С другой стороны, группа \mathcal{G} действует транзитивно на элементах из $B^{\mathbb{F}_q\mathcal{G}}$ и не нарушает M -ортогональности. Это позволяет построить набор декодирующих графов для всех координат только по одному такому графу. Отметим, что в разд. 2 рассматривалось декодирование только одной выбранной координаты по соответствующему этой координате декодирующему графу, и в общем случае построение декодирующего графа для каждой отдельной координаты — независимая задача. Однако в случае групповых кодов, имея декодирующий граф для одной координаты, можно построить декодирующие графы для каждой координаты благодаря алгебраической структуре групповых кодов.

Пусть \mathcal{G} и \mathcal{H} — конечные группы мощности n_1 и n_2 соответственно, а $\mathbb{F}_q\mathcal{G}$, $\mathbb{F}_q\mathcal{H}$ — их групповые алгебры. Рассмотрим два M -ортогональных семейства групповых кодов

$$\begin{aligned}\mathcal{S}_1 &= \{C_1(0), C_1(1), \dots, C_1(J)\}, & C_1(i) &\in \mathbb{F}_q\mathcal{G}, \\ \mathcal{S}_2 &= \{C_2(0), C_2(1), \dots, C_2(J)\}, & C_2(i) &\in \mathbb{F}_q\mathcal{H}.\end{aligned}\tag{10}$$

Пусть $C(D) \in \mathbb{F}_q(\mathcal{G} \times \mathcal{H})$ — код длины n_1n_2 , построенный на основе этих семейств (см. (4), леммы 1 и 2). Тогда двойственный к нему код $\overline{C}(D)$ называется *групповым D -кодом*. Далее будем рассматривать только групповые D -коды. Отметим, что тензорное произведение групповых MLD-кодов входит в класс групповых D -кодов и разработанные ниже алгоритмы подходят для их декодирования.

Пусть $\mathbf{c} \in \overline{C}(D)$ — кодовый вектор, который на выходе из канала принимает вид

$$\mathbf{x} = \mathbf{c} + \mathbf{e}, \quad \mathbf{e} \in \mathbb{F}_q(\mathcal{G} \times \mathcal{H}).\tag{11}$$

Ниже построим алгоритмы, позволяющие находить значение вектора ошибок \mathbf{e} при условии, что $\text{wt}_B(\mathbf{e}) \leq \lfloor (d-1)/2 \rfloor$, где $d = d_{B^{\mathbb{F}_q(\mathcal{G} \times \mathcal{H})}}(\overline{C}(D))$ (см. (9)) — минимальное кодовое расстояние кода $C(D)$, определяемое в соответствии с леммой 2. В п. 4.2 строится кодирующая матрица для D -кода и тем самым решается задача кодирования, а задача декодирования решается в несколько этапов. В п. 4.3 по декодирующим графам $G(\mathcal{S}_1, \mathbf{b}^1)$, $G(\mathcal{S}_2, \mathbf{b}^2)$ двух семейств MLD-кодов строится декодирующий граф $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ для D -кода, а в п. 4.4 с помощью построенного графа выполняется декодирование координаты, соответствующей

вектору $\mathbf{b}^1 \otimes \mathbf{b}^2$. Наконец, в п. 4.5 приводится алгоритм построения набора декодирующих графов для группового D -кода и алгоритм декодирования по этому набору всех координат зашумлённого кодового слова.

4.2. Конструкция кодера. Рассмотрим два семейства групповых кодов $\mathcal{S}_1, \mathcal{S}_2$ (см. (10)), которые удовлетворяют условиям (s1)–(s4), и множество $D \subset D_0$. Пусть по множеству D построены соответствующие множества D^*, D_b, D_b^* из п. 1.2. Тогда по лемме 3 имеем

$$\overline{C}(D) = L\left(\bigcup_{(i,j) \in D_b^*} \overline{C}_1(i) \otimes \overline{C}_2(j)\right). \quad (12)$$

Пусть $G_{\overline{C}_t(i)}$ — порождающая матрица кода $\overline{C}_t(i)$, двойственного коду $C_t(i) \in \mathcal{S}_t$, $t \in \{1, 2\}$. Зафиксируем линейный порядок

$$(i_1, j_1), (i_2, j_2), \dots, (i_{|D_b^*|}, j_{|D_b^*|})$$

на множестве D_b^* и построим матрицу

$$\widetilde{G} = \begin{pmatrix} G_{\overline{C}_1(i_1)} \otimes G_{\overline{C}_2(j_1)} \\ G_{\overline{C}_1(i_2)} \otimes G_{\overline{C}_2(j_2)} \\ \dots \\ G_{\overline{C}_1(i_{|D_b^*|})} \otimes G_{\overline{C}_2(j_{|D_b^*|})} \end{pmatrix}.$$

Пусть $G_{\overline{C}(D)}$ — $(k \times n)$ -матрица полного ранга, эквивалентная матрице \widetilde{G} . Тогда $G_{\overline{C}(D)}$ — порождающая матрица кода $\overline{C}(D)$, где длина кода есть $n = n_1 n_2$, а размерность в соответствии с леммой 1 равна

$$k = n_1 n_2 - \dim C(D) = n_1 n_2 - \sum_{(i,j) \in D} k_1(i) k_2(j).$$

Процедура кодирования информационного сообщения $a \in \mathbb{F}_q^k$ заключается в умножении его справа на матрицу $G_{\overline{C}(D)}$.

4.3. Построение декодирующего графа. Опишем процедуру построения декодирующего графа для кода $\overline{C}(D)$ (см. (12)). Из [4, с. 121–122] (см. также [11, лемма 1]) вытекает

Лемма 4. Пусть $\mathbf{c}_t \in \mathbb{F}^{n_t}$ и $\mathcal{M}_{\mathbf{c}_t}$ — M -ортогональное множество для вектора \mathbf{c}_t , $t \in \{1, 2\}$. Тогда M -ортогональное множество для вектора $\mathbf{c}_1 \otimes \mathbf{c}_2$ состоит из векторов следующих видов:

- 1) $\mathbf{c}_1 \otimes \mathbf{w}$, $\mathbf{w} \in \mathcal{M}_{\mathbf{c}_2}$,
- 2) $\mathbf{v} \otimes \mathbf{c}_2$, $\mathbf{v} \in \mathcal{M}_{\mathbf{c}_1}$,
- 3) $\mathbf{c}_1 \otimes \mathbf{w} + \mathbf{v} \otimes \mathbf{c}_2 + \mathbf{v} \otimes \mathbf{w}$, $\mathbf{v} \in \mathcal{M}_{\mathbf{c}_1}$, $\mathbf{w} \in \mathcal{M}_{\mathbf{c}_2}$.

В разд. 2 разработана конструкция декодирующего графа $G(\mathcal{S}, \mathbf{b}) = (V, E)$ для произвольного M -ортогонального семейства \mathcal{S} , содержащая условия (g1)–(g4). Пусть $G_1(\mathcal{S}_1, \mathbf{b}^1) = (V_1, E_1)$, $G_2(\mathcal{S}_2, \mathbf{b}^2) = (V_2, E_2)$ — декодирующие графы для семейств $\mathcal{S}_1, \mathcal{S}_2$ соответственно. Ниже будем использовать обозначения из (g1)–(g4), адаптированные к семействам $\mathcal{S}_1, \mathcal{S}_2$, т. е. заменять \mathbf{b} на \mathbf{b}^t , \mathcal{S} на \mathcal{S}_t , где $t \in \{1, 2\}$, и добавлять при необходимости к обозначениям из (g1)–(g4) справа сверху индекс t , соответствующий семейству \mathcal{S}_t . По этим графам определим граф $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2) = (V^\otimes, E^\otimes)$, с помощью которого выполняется декодирование кода $\overline{C}(D)$.

1) Вершины графа будем помечать одним из трёх типов типа вектора в соответствии с леммой 4, а также некоторыми дополнительными метками. А именно, вершины первого и второго типов будем дополнительно помечать такой тройкой: вектором $\mathbf{x}_k \otimes \mathbf{x}_l$ и парой вершин (v_k^1, v_l^2) , где $v_k^1(\mathbf{x}_k, i) \in V_1$, $v_l^2(\mathbf{x}_l, j) \in V_2$, а $\mathbf{c}_1 \in C_1(i)$, $\mathbf{c}_2 \in C_2(j)$. Вершины третьего типа будем дополнительно помечать вектором $\mathbf{c} \in \mathbb{F}(\mathcal{G} \times \mathcal{H})$.

2) В графе существует единственный исток, который помечен набором $(\mathbf{b}^1 \otimes \mathbf{b}^2, v_0^1(\mathbf{b}^1, 0), v_0^2(\mathbf{b}^2, 0), \text{type})$, где $\text{type} = 1$, а v_0^t — исток графа G_t семейства \mathcal{S}_t .

3) Стоки графа помечены наборами $(\mathbf{x}_k^1 \otimes \mathbf{x}_k^2, v^1(\mathbf{x}_k^1, i), v^2(\mathbf{x}_k^2, j), \text{type})$ такими, что $(i, j) \in D_b$.

4) Любая вершина $v(\mathbf{x}_k^1 \otimes \mathbf{x}_k^2, v^1(\mathbf{x}_k^1, i), v^2(\mathbf{x}_k^2, j), \text{type})$ с $\text{type} \in \{1, 2\}$, не являющаяся стоком графа, будет началом $d_i d_j - 1$ ориентированных дуг (d_i и d_j определяются в соответствии с условием (g4)) с концами в таких вершинах, что векторы, которыми помечены эти вершины, составляют M -ортогональное множество для вектора $\mathbf{x}_k^1 \otimes \mathbf{x}_k^2$.

Отметим, что из любой вершины существует путь к одному из стоков графа. Вершины графа обозначим именем вершины и списком всех меток, при этом длина списка зависит от типа. Кроме того, будем использовать сокращённые обозначения $v(\mathbf{x}_k^1 \otimes \mathbf{x}_k^2)$ или просто v .

Сконструируем алгоритмы `build_tens_node` и `build_tens_graph`, реализующие процедуру построения декодирующего графа для кода $\overline{C}(D)$.

Алгоритм 4 (`build_tens_node`)

Вход: $G_t(\mathcal{S}_t, \mathbf{b}^t) = (V_t, E_t)$ — декодирующий граф для семейства \mathcal{S}_t , $t = 1, 2$, $v_z(\mathbf{x}_k^1 \otimes \mathbf{x}_k^2, v_k^1, v_k^2, \text{type}) \in V^\otimes$, $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ — промежуточный декодирующий граф для кода $\overline{C}(D)$

Выход: $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ — следующая промежуточная версия декодирующего графа для кода $\overline{C}(D)$

```

1: Desc1 := descendants( $v_k^1(\mathbf{x}_k^1, (i^1, j^1))$ ,  $G(\mathcal{S}_1)$ )
2: Desc2 := descendants( $v_k^2(\mathbf{x}_k^2, (i^2, j^2))$ ,  $G(\mathcal{S}_1)$ )
3: for all  $v_m^2(\mathbf{x}_m^2, (j_2, p_2)) \in \text{Desc}_2$  do
4:   if  $v_r(\mathbf{x}_k^1 \otimes \mathbf{x}_m^2, v_k^1, v_m^2, 1) \notin V^\otimes$  then
5:     добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  вершину  $v_r(\mathbf{x}_k^1 \otimes \mathbf{x}_m^2, v_k^1, v_m^2, 1)$ 
6:   end if
7:   добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  дугу  $(v_z, v_r)$ 
8:   if  $(i_1, j_2) \notin D$  then
9:     build_tens_node( $G_1, G_2, v_r, G(\mathcal{S}_1 \otimes \mathcal{S}_2)$ )
10:  end if
11: end for
12: for all  $v_l^1(\mathbf{x}_l^1, (j_1, p_1)) \in \text{Desc}_1$  do
13:   if  $v_r(\mathbf{x}_l^1 \otimes \mathbf{x}_k^2, v_l^1, v_k^2, 2) \notin V^\otimes$  then
14:     добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  вершину  $v_r(\mathbf{x}_l^1 \otimes \mathbf{x}_k^2, v_l^1, v_k^2, 2)$ 
15:   end if
16:   добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  дугу  $(v_z, v_r)$ 
17:   if  $(j_1, i_2) \notin D$  then
18:     build_tens_node( $G_1, G_2, v_r, G(\mathcal{S}_1 \otimes \mathcal{S}_2)$ )
19:   end if
20: end for
21: for all  $v_m^2(\mathbf{x}_m^2, (j_2, p_2)) \in \text{Desc}_2$  do
22:   for all  $v_l^1(\mathbf{x}_l^1, (j_1, p_1)) \in \text{Desc}_1$  do
23:     добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  вершину
       $v_{t3}(\mathbf{x}_k^1 \otimes \mathbf{x}_m^2 + \mathbf{x}_l^1 \otimes \mathbf{x}_k^2 + \mathbf{x}_l^1 \otimes \mathbf{x}_m^2, 3)$ 
24:     добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  дугу  $(v_z, v_{t3})$ 
25:     if  $\mathbf{x}_{t3} \notin C(D)$  then
26:       добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  дугу  $(v_{t3}, v_{z1}(\mathbf{x}_k^1 \otimes \mathbf{x}_m^2))$ 
         $\triangleright v_{z1} \in V^\otimes$  — вершина графа  $G$ , помеченная
        вектором  $\mathbf{x}_k^1 \otimes \mathbf{x}_m^2$ 
27:       добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  дугу  $(v_{t3}, v_{z2}(\mathbf{x}_l^1 \otimes \mathbf{x}_k^2))$ 
         $\triangleright v_{z2} \in V^\otimes$  — вершина графа  $G$ , помеченная
        вектором  $\mathbf{x}_l^1 \otimes \mathbf{x}_k^2$ 
28:       добавить в граф  $G(\mathcal{S}_1 \times \mathcal{S}_2)$  дугу  $(v_{t3}, v_{z3}(\mathbf{x}_l^1 \otimes \mathbf{x}_m^2))$ 
         $\triangleright v_{z3} \in V^\otimes$  — вершина графа  $G$ , помеченная
        вектором  $\mathbf{x}_l^1 \otimes \mathbf{x}_m^2$ 
29:     end if
30:   end for
31: end for
32: return  $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ 

```

Алгоритм `build_tens_node` получает на вход графы G_1, G_2 , соответствующие семействам $\mathcal{S}_1, \mathcal{S}_2$ (см. разд. 2), промежуточную версию итогового графа $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ с текущей вершиной этого графа и возвращает следующую промежуточную версию этого графа.

Пусть текущая вершина первого или второго типа. Метки вершин исходных графов, которыми помечена текущая вершина, соответствуют точкам множества D_0 . Алгоритм выполняет свою работу для точек, не принадлежащих множеству D . Для текущей вершины находятся множества потомков для вершин исходных графов G_1 и G_2 , которыми помечена текущая. Далее, в соответствии с леммой 4 для текущей вершины сначала строится множество потомков первого типа, при этом после нахождения каждого такого потомка вершина добавляется к множеству вершин итогового графа, добавляется дуга, соединяющая текущую вершину с потомком, и если новая вершина не принадлежит D_b и тем более D , то выполняется рекурсивный вызов алгоритма уже для потомка текущей вершины.

После добавления в граф потомков первого типа происходит вычисление потомков второго типа и выполняются аналогичные действия.

В заключение строятся потомки третьего типа, каждый из которых соединяется дугами с вершинами первого и второго типов. Каждая вершина третьего типа помечена вектором, который является суммой трёх векторов. При добавлении исходящих дуг из вершины третьего типа выполняется поиск вершин первого или второго типов, которые помечены этими векторами, и найденные вершины будут концами этих дуг.

Как было указано ранее, каждая вершина графа соответствует точке множества D_0 . При добавлении в граф вершин проверяется принадлежность соответствующей точки множеству D . Алгоритм `build_tens_node` заканчивает свою работу после добавления всех вершин, соответствующих точкам множества D_b , т. е. стоков.

Алгоритм 5 (`build_tens_graph`)

Вход: $G_t(\mathcal{S}_t, \mathbf{b}^t) = (V_t, E_t)$ — декодирующий граф для семейства \mathcal{S}_t , $v_0^t(\mathbf{b}^t, (0, i^t)) \in V_t$ — исток графа G_t , $t = 1, 2$

Выход: $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ — декодирующий граф для кода $\overline{C}(D)$

- 1: $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2) := \emptyset$
 - 2: добавить в граф G вершину $v_0(\mathbf{b}^1 \otimes \mathbf{b}^2, v_0^1, v_0^2, 1)$
 - 3: $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2) := \text{build_tens_node}(G_1, G_2, v_0, G)$
 - 4: **return** $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$
-

Алгоритм `build_tens_graph` принимает на вход декодирующие графы G_1 и G_2 для семейств \mathcal{S}_1 и \mathcal{S}_2 , а также истоки этих графов и строит

декодирующий граф $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ для кода $\overline{C}(D)$. В алгоритме после инициализации искомого графа добавляется исток и выполняется обращение к алгоритму `build_tens_node`, на вход которому подаётся этот исток. Результатом работы алгоритма является граф для декодирования координаты кода $\overline{C}(D)$, соответствующей вектору $\mathbf{b}^1 \otimes \mathbf{b}^2$.

4.4. Декодирование одной координаты. Выше построен декодирующий граф $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ для кода $\overline{C}(D)$. Теперь построим алгоритмы декодирования с помощью этого графа: `decode_tens_node` и `decode_tens_bit`. Как и в алгоритме из разд. 2, будем пользоваться вспомогательным алгоритмом `descendants`, который возвращает множество потомков заданной вершины. Также будем пользоваться вспомогательным алгоритмом `type`, возвращающим тип заданной вершины.

Алгоритм 6 (`decode_tens_node`)

Вход: $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ — декодирующий граф для кода $\overline{C}(D)$, $v_r \in G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$, \mathbf{y} — принятый по каналу вектор

Выход: $l(v_r) \in \mathbb{F}_q$ — значение метки для вершины v_r

```

1: Desc := descendants( $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2), v_r$ )
2: if Desc =  $\{\emptyset\}$  then
3:    $l(v_r) := (\mathbf{x}_r, \mathbf{y})$ 
4: else if type( $v_r$ ) = 1 or type( $v_r$ ) = 2 then
5:    $M_l := \emptyset$ 
6:   for all  $v_l \in \text{Desc}$  do
7:     if  $l(v_l) \neq \emptyset$  then
8:       к  $M_l$  добавить  $l(v_l)$ 
9:     else
10:      к  $M_l$  добавить decode_tens_node( $G, v_l, \mathbf{y}$ )
11:   end if
12: end for
13:  $l(v_r) := \text{MajorVote}(M_l)$ 
14: else
15:    $S := 0$ 
16:   for all  $v_l \in \text{Desc}$  do
17:      $S := S + \text{decode_tens_node}(G, v_l, \mathbf{y}) \bmod q$ 
18:   end for
19:    $l(v_r) := S$ 
20: end if
21: пометить вершину  $v_r$  меткой  $l(v_r)$ 
22: return  $l(v_r)$ 

```

Алгоритм 7 (decode_tens_bit)

Вход: \mathbf{y} — зашумлённый кодовый вектор, $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ — декодирующий граф для кода $\overline{C}(D)$, $v_0(\mathbf{b}^1 \otimes \mathbf{b}^2)$ — исток графа G

Выход: $\tilde{\mathbf{x}} \in \mathbb{F}_q^n$ — зашумлённый кодовый вектор с декодированной координатой, отвечающей вектору, которым помечен исток v_0 графа G

- 1: $l := \text{decode_tens_node}(G, v_0, \mathbf{y})$
- 2: $\tilde{\mathbf{x}} = \mathbf{y} + \mathbf{b}_l$ $\triangleright \mathbf{b}_l$ — вектор $\mathbf{b} = \mathbf{b}^1 \otimes \mathbf{b}^2$, в котором ненулевая координата заменена на l
- 3: **return** $\tilde{\mathbf{x}}$

Первый из этих алгоритмов — `decode_tens_node` — помечает вершину v_r графа $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$ элементом из \mathbb{F}_q . Эти метки будем называть декодирующими метками. Вычисление меток выполняется по следующему правилу. Для вершин первого и второго типов, не принадлежащих двойственному коду, значение декодирующей метки получается голосованием по меткам потомков с помощью алгоритма `MajorVote`. Для вершин третьего типа, не принадлежащих двойственному коду, вычисляется сумма декодирующих меток потомков. Для вершин, помеченных векторами из двойственного кода, значения декодирующих меток получаются вычислением скалярного произведения векторов, которыми помечены эти вершины, с зашумлённым кодовым вектором. Результатом работы алгоритма является значение декодирующей метки для вершины v_r .

Алгоритм `decode_tens_bit` декодирует координату зашумлённого вектора, соответствующую базисному вектору $\mathbf{b}^1 \otimes \mathbf{b}^2$. Сначала с помощью алгоритма `decode_tens_node` вычисляется значение метки для истока графа $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}^1 \otimes \mathbf{b}^2)$. После чего в соответствии со значением этой метки декодируется координата зашумлённого вектора.

Алгоритм 8 (CloneGraph)

Вход: \mathcal{G} , $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b})$ — декодирующий граф для кода $\overline{C}(D)$ и координаты, соответствующей вектору \mathbf{b} , вектор \mathbf{b}'

Выход: \mathcal{G} , $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}')$

- 1: $\mathcal{G}, G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}') := \mathcal{G}, G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b})$
- 2: найти $g \in \mathcal{G}$ такой, что $(g, \mathbf{b}) = \mathbf{b}'$
- 3: **for all** метки \mathbf{p} графа $\mathcal{G}, G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}')$ **do**
- 4: $\mathbf{p} := (g, \mathbf{p})$ \triangleright действие элементом g на \mathbf{p} по правилу (7)
- 5: **end for**
- 6: **return** $\mathcal{G}, G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}')$

4.5. Декодирование кодового слова. В [17] для групповых кодов построен алгоритм CloneTree, позволяющий по декодирующему дереву с одной меткой у корня построить декодирующее дерево для корня с любой другой меткой. Приведём формальное описание нового алгоритма CloneGraph, который является модификацией алгоритма CloneTree на случай использования декодирующего графа. Алгоритм CloneGraph по графу $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b})$ строит граф $G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}')$, где $\mathbf{b}, \mathbf{b}' \in B^{\mathbb{F}_q(\mathcal{G} \times \mathcal{H})}$.

Таким образом, набор декодирующих графов

$$G(\mathcal{S}_1 \times \mathcal{S}_2) = \{G(\mathcal{S}_1 \times \mathcal{S}_2, \delta_{(g,h)})\}_{\delta_{(g,h)} \in B^{\mathbb{F}_q(\mathcal{G} \times \mathcal{H})}}$$

для группового кода $\overline{C}(D)$ может быть построен по одному графу, а именно, для любой базисной функции $\delta_{(g,h)} \in B^{\mathbb{F}_q(\mathcal{G} \times \mathcal{H})}$ имеем

$$G(\mathcal{S}_1 \times \mathcal{S}_2, \delta_{(g,h)}) = \text{CloneGraph}(\mathcal{G} \times \mathcal{H}, G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{1}_{\mathcal{G}} \otimes \mathbf{1}_{\mathcal{H}}), \delta_{(g,h)}).$$

Алгоритм 9 (decode_tensor_vector)

Вход: $\mathbf{x} = \mathbf{c} + \mathbf{e}$ — принятый вектор, $G(\mathcal{S}_1 \times \mathcal{S}_2)$ — набор декодирующих графов

Выход: вектор \mathbf{c}' — результат декодирования

- 1: **for all** $\mathbf{b} \in B^{\mathbb{F}_q(\mathcal{G} \times \mathcal{H})}$ **do**
 - 2: $x_{\mathbf{b}} := x_{\mathbf{b}} - \text{decode_tens_bit}(\mathbf{x}, G(\mathcal{S}_1 \times \mathcal{S}_2, \mathbf{b}))$
 - 3: **end for**
 - 4: **return** \mathbf{c}'
-

Алгоритм decode_tensor_vector декодирования зашумлённого кодового вектора \mathbf{x} , в котором каждая координата декодируется с помощью алгоритма decode_tens_bit, построен по аналогии с алгоритмом Decoder3 из [17].

ЛИТЕРАТУРА

1. Massey J. L. Threshold decoding. Cambridge, MA: MIT Press, 1963.
2. Кларк Дж. мл., Кейн Дж. Кодирование с исправлением ошибок в системах цифровой связи. М.: Радиосвязь, 1981.
3. Сидельников В. М. Открытое шифрование на основе двоичных кодов Рида — Маллера // Дискрет. математика. 1994. Т. 6, № 2. С. 3–20.
4. Циммерман К.-Х. Методы теории модулярных представлений в алгебраической теории кодирования. М.: МЦНМО, 2011.
5. NIST reveals 26 algorithms advancing to the post-quantum crypto ‘Semifinals’ // National Institute of Standards and Technology. Available at <http://www.nist.gov/news-events/news/2019/01/nist-reveals-26-algorithms-advancing-post-quantum-crypto-semifinals> (accessed May 23, 2020).

6. **Borodin M. A., Chizhov I. V.** Effective attack on the McEliece cryptosystem based on Reed–Muller codes // *Discrete Math. Appl.* 2014. Vol. 26, No. 1. P. 273–280.
7. **Sidel'nikov V. M., Shestakov S. O.** On an encoding system constructed on the basis of generalized Reed–Solomon codes // *Discrete Math. Appl.* 1992. Vol. 2, No. 4. P. 439–444.
8. **Minder L., Shokrollahi A.** Cryptanalysis of the Sidelnikov cryptosystem // *Advances in Cryptology – EUROCRYPT 2007. Proc. 26th Annu. Int. Conf. Theory Appl. Cryptogr. Tech. (Barcelona, Spain, May 20–24, 2007)*. Heidelberg: Springer, 2007. P. 347–360. (Lect. Notes Comput. Sci.; Vol. 4515).
9. **Wieschebrink C.** Cryptanalysis of the Niederreiter public key scheme based on GRS subcodes // *Post-Quantum Cryptography. Proc. 3rd Int. Workshop (Darmstadt, Germany, May 25–28, 2010)*. Heidelberg: Springer, 2010. P. 61–72. (Lect. Notes Comput. Sci.; Vol. 6061).
10. **Деундяк В. М., Косолапов Ю. В.** Криптосистема на индуцированных групповых кодах // *Моделирование и анализ информ. систем.* 2016. Т. 23, № 2. С. 137–152.
11. **Деундяк В. М., Косолапов Ю. В., Лелюк Е. А.** Декодирование тензорного произведения MLD-кодов и приложения к кодовым криптосистемам // *Моделирование и анализ информ. систем.* 2017. Т. 24, № 2. С. 239–252.
12. **Kasami T., Lin S.** On the construction of a class of majority-logic decodable codes // *IEEE Trans. Inform. Theory.* 1971. Vol. IT-17, No. 5. P. 600–610.
13. **Сидельников В. М.** Теория кодирования. М.: ФИЗМАТЛИТ, 2008.
14. **Morelos-Zaragoza R. H.** The art of error correcting coding. Chichester: Wiley, 2006.
15. **Блох Э. Л., Зяблов В. В.** Кодирование обобщённых каскадных кодов // *Пробл. передачи информации.* 1974. Т. 10, № 3. С. 45–50.
16. **Зиновьев В. А.** Обобщённые каскадные коды // *Пробл. передачи информ.* 1976. Т. 12, № 1. С. 5–15.
17. **Деундяк В. М., Косолапов Ю. В.** Алгоритмы для мажоритарного декодирования групповых кодов // *Моделирование и анализ информ. систем.* 2015. Т. 22, № 4. С. 464–482.
18. **Curtis C. W., Reiner I.** Representation theory of finite groups and associative algebras. New York: Interscience, 1962.

Деундяк Владимир Михайлович
Лелюк Евгений Андреевич

Статья поступила
24 февраля 2019 г.
После доработки —
17 октября 2019 г.
Принята к публикации
27 ноября 2019 г.

A GRAPH-THEORETICAL METHOD
FOR DECODING SOME GROUP MLD-CODESV. M. Deundyak^{1,2,a} and E. A. Lelyuk^{1,b}¹ Vorovich Institute of Mathematics, Mechanics, and Computer Science,
8a Milchakov Street, 344058 Rostov-on-Don, Russia² Scientific and Research Institute “Spetsvuzavtomatika”,
51 Gazetnyi Lane, 344002 Rostov-on-Don, RussiaE-mail: ^av1.deundyak@gmail.com, ^blelukevgeniy@mail.ru

Abstract. We construct the class of majority-logical decodable group codes using a method for combining the codes that are based on the tensor product and the sum of codes. The construction of this class rests on the Kasami–Lin technique, which allows us to consider not only individual codes but also families of codes and utilizes the M -orthogonality construction presented by Massey that is important for the majority-logical decodable codes. The codes under study are ideals in group algebras over, generally speaking, noncommutative finite groups. Some algorithmic model of the majority decoding for the codes under consideration is developed that is based on the graph-theoretic approach. An important part of this model is the construction of a special decoding graph for decoding one coordinate of a noisy codeword corresponding to this graph. The group properties of the codes enable us to quickly find decoding graphs for the remaining coordinates. We develop some decoding algorithm that corrects the errors in all coordinates of the noisy codeword using this decoding graphs. As an example of families of group codes, we give the Reed–Muller binary codes important in cryptography. The code cryptosystems are considered as an alternative to the number-theoretic cryptosystems widely used at present since they are resistant to attacks by quantum computers. The relevance of the problem under consideration lies in the fact that the use of group codes and their various combinations is currently one of the promising ways to increase the stability of code cryptosystems because enables us to construct new codes with a complex algebraic structure, which positively affects the stability of the code cryptosystems. Illustr. 2, bibliogr. 18.

Keywords: MLD-code, majority decoding, group code, tensor product, graph.

REFERENCES

1. **J. L. Massey**, *Threshold Decoding*, (MIT Press, Cambridge, 1963).
2. **G. C. Clark, Jr.** and **J. B. Cain**, *Error-Correction Coding for Digital Communications* (Plenum Press, New York, 1981; Radiosvyaz', Moscow, 1981 [Russian]).
3. **V. M. Sidel'nikov**, Open encryption based on binary Reed–Muller codes, *Diskretn. Mat.* **6** (2), 3–20 (1994) [Russian].
4. **K.-Kh. Tsimmerman**, *Methods of the Modular Representations Theory in Algebraic Coding Theory* (MTsNMO, Moscow, 2011) [Russian].
5. NIST reveals 26 algorithms advancing to the post-quantum crypto ‘Semifinals’. Available at <http://www.nist.gov/news-events/news/2019/01/nist-reveals-26-algorithms-advancing-post-quantum-crypto-semifinals> (accessed May 23, 2020).
6. **M. A. Borodin** and **I. V. Chizhov**, Effective attack on the McEliece cryptosystem based on Reed–Muller codes, *Discrete Math. Appl.* **26** (1), 273–280 (2014).
7. **V. M. Sidel'nikov** and **S. O. Shestakov**, On an encoding system constructed on the basis of generalized Reed–Solomon codes, *Discrete Math. Appl.* **2** (4), 439–444 (1992).
8. **L. Minder** and **A. Shokrollahi**, Cryptanalysis of the Sidelnikov cryptosystem, *Advances in Cryptology – EUROCRYPT 2007* (Proc. 26th Annu. Int. Conf. Theory Appl. Cryptogr. Tech., Barcelona, Spain, May 20–24, 2007) (Springer, Heidelberg, 2007), pp. 347–360 (Lect. Notes Comput. Sci., Vol. 4515).
9. **C. Wieschebrink**, Cryptanalysis of the Niederreiter Public Key Scheme Based on GRS Subcodes *Post-Quantum Cryptography* (Proc. 3rd Int. Workshop, Darmstadt, Germany, May 25–28, 2010) (Springer, Heidelberg, 2010), pp. 61–72 (Lect. Notes Comput. Sci., Vol. 6061).
10. **V. M. Deundyak** and **Yu. V. Kosolapov**, Cryptosystem based on induced group codes, *Model. Anal. Inf. Sist.* **23** (2), 137–152 (2016) [Russian].
11. **V. M. Deundyak**, **Yu. V. Kosolapov**, and **E. A. Lelyuk**, Decoding the tensor product of MLD codes and applications for code cryptosystems, *Model. Anal. Inf. Sist.* **24** (2), 239–252 (2017) [Russian] [*Autom. Control Comput. Sci.* **52** (7), 647–657 (2018)].
12. **T. Kasami** and **S. Lin**, On the construction of a class of majority-logic decodable codes, *IEEE Trans. Inf. Theory* **IT-17** (5), 600–610 (1971).
13. **V. M. Sidel'nikov**, *Coding Theory* (FIZMATLIT, Moscow, 2008) [Russian].
14. **R. H. Morelos-Zaragoza**, *The Art of Error Correcting Coding* (Wiley, Chichester, 2006).
15. **Eh. L. Blokh** and **V. V. Zyablov**, Coding by generalized concatenated codes, *Probl. Peredachi Inf.* **10** (3), 45–50 (1974) [Russian] [*Probl. Inf. Transm.* **10** (3), 218–222 (1974)].

16. **V. A. Zinov'ev**, Generalized concatenated codes, *Probl. Peredachi Inf.* **12** (1), 5–15 (1976) [Russian].
17. **V. M. Deundyak** and **Y. V. Kosolapov**, Algorithms for majority decoding of group codes, *Model. Anal. Inf. Sist.* **22** (4), 464–482 (2015) [Russian].
18. **C. W. Curtis** and **I. Reiner**, *Representation Theory of Finite Groups and Associative Algebras* (Interscience, New York, 1962).

Vladimir M. Deundyak
Evgeny A. Lelyuk

Received February 24, 2019
Revised October 17, 2019
Accepted November 27, 2019