Math-Net.Ru

Общероссийский математический портал

В. Л. Куракин, Алгоритм Берлекэмп–Месси над конечными кольцами, модулями и бимодулями, *Дискрет. матем.*, 1998, том 10, выпуск 4, 3–34

DOI: <https://doi.org/10.4213/dm446>

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением
<http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 54.84.234.179

26 апреля 2023 г., 12:35:31

УДК 519.7

Алгоритм Берлекэмпа–Месси над конечными кольцами, модулями и бимодулями

© 1998 г. В. Л. Куракин

Предлагается алгоритм построения унитарного многочлена наименьшей степени, порождающего заданный отрезок $u(0, l-1) = (u(0), u(1), \dots, u(l-1))$ длины l с трудоемкостью $O(l^2)$ операций при $l \rightarrow \infty$. Рассматриваются случаи, когда $u(0, l-1)$ — отрезок над конечным кольцом R с единицей, над конечным модулем ${}_R M$ или над конечным бимодулем ${}_A M_B$, где A и B — конечные кольца с единицами.

1. Введение

Алгоритм Берлекэмпа–Месси для отрезков над полем впервые представлен в работах [7, 73] и затем исследовался и усовершенствовался многими авторами. Этот алгоритм находит унитарный многочлен наименьшей степени, порождающий заданный отрезок длины l над полем, с трудоемкостью $O(l^2)$ операций. Алгоритм Берлекэмпа–Месси используется в теории кодирования при декодировании кодов Рида–Соломона, кодов БЧХ, циклических и полициклических кодов, альтернантных кодов, кодов Гоппы, алгебро-геометрических кодов. Он также используется при решении ганкелевых, теплицевых и разреженных систем линейных уравнений, в задачах рациональной интерполяции и аппроксимации Паде, при поиске минимальных реализаций линейных систем, при тестировании псевдослучайных последовательностей, для быстрого вычисления в конечных полях, а также в криптографии. Краткий обзор литературы по алгоритму Берлекэмпа–Месси приводится в конце работы.

Известны обобщения алгоритма Берлекэмпа–Месси для отрезков над различными кольцами. Алгоритм Берлекэмпа–Месси над кольцами матриц над полем рассматривался в [27, 31, 62, 63, 68, 112], причем в [27, 62, 63] рассматривался вероятностный алгоритм. В [85] рассматривается алгоритм Берлекэмпа–Месси над кольцами вычетов $\mathbf{Z}_m$, в [93] приводится его обобщение на nD -последовательности. В [45, 47, 79] приводится алгоритм Берлекэмпа–Месси над коммутативной областью целостности с однозначным разложением на множители. В [1] алгоритм Берлекэмпа–Месси обобщен на случай произвольного конечного коммутативного кольца с единицей. В данной работе рассматривается еще более общая ситуация, когда исходный отрезок является отрезком над конечным (не обязательно коммутативным) кольцом с единицей, над конечным модулем или над конечным бимодулем.

Основной алгоритм излагается для отрезка над конечным левым модулем ${}_R M$, где R — конечное кольцо с единицей. Понятно, что для отрезка над конечным правым модулем M_R алгоритм будет аналогичным. В частном случае $M = R$ получаем алгоритм для отрезка над конечным кольцом R . Случай отрезка над конечным бимодулем ${}_A M_B$, где A и B — конечные кольца с единицами, сводится к случаю отрезка над модулем ${}_R M$, где $R = A \otimes B^{op}$ — тензорное произведение колец A и B^{op} над $\mathbf{Z}$, а B^{op} — кольцо, противоположное B . В параграфах 2 и 3 излагается и обосновывается алгоритм Берлекэмп–Мессе для отрезка над левым модулем ${}_R M$, в параграфе 4 рассматривается случай отрезка над кольцом или над бимодулем. В параграфе 5 приводится пример, а параграф 6 посвящен краткому обзору литературы. В этом первом параграфе вводятся необходимые определения.

Пусть R — конечное коммутативное кольцо с единицей e , ${}_R M$ — левый R -модуль (см. [5]), $|M| < \infty$, и пусть $u(0, l-1) = (u(0), u(1), \dots, u(l-1)) \in M^l$ — отрезок длины l над модулем M . Будем говорить, что унитарный многочлен

$$f(x) = x^s - c_{s-1}x^{s-1} - \dots - c_1x - c_0 \in R[x]$$

порождает отрезок $u(0, l-1)$, если $s \geq l$ или если $s < l$ и

$$u(i+s) = c_{s-1}u(i+s-1) + \dots + c_1u(i+1) + c_0u(i), \quad i = 0, 1, \dots, l-s-1. \quad (1)$$

Унитарный многочлен наименьшей степени, порождающий отрезок $u(0, l-1)$, назовем минимальным многочленом, а степень этого многочлена назовем рангом отрезка $u(0, l-1)$. Алгоритм Берлекэмп–Мессе находит минимальный многочлен отрезка $u(0, l-1)$ с трудоемкостью $O(l^2)$ операций при $l \rightarrow \infty$. Излагаемый ниже алгоритм является модификацией алгоритма работы [1] для случая отрезка над модулем.

Суммой отрезков $u(0, l_1-1) \in M^{l_1}$ и $v(0, l_2-1) \in M^{l_2}$ назовем отрезок $w(0, l-1) \in M^l$ длины $l = \min\{l_1, l_2\}$ вида

$$w(0, l-1) = u(0, l_1-1) + v(0, l_2-1) = (u(0) + v(0), \dots, u(l-1) + v(l-1)).$$

Сдвигом отрезка $u(0, l-1)$ влево на s шагов, $s = 0, 1, \dots, l-1$, будем называть отрезок

$$x^s u(0, l-1) = (u(s), \dots, u(l-1))$$

над модулем M длины $l-s$.

Произведением элемента $r \in R$ на отрезок $u(0, l-1)$ назовем отрезок

$$ru(0, l-1) = (ru(0), \dots, ru(l-1)).$$

Произведением многочлена $f(x) = f_s x^s + \dots + f_1 x + f_0 \in R[x]$ степени $s < l$ на отрезок $u(0, l-1)$ будем называть отрезок $v(0, l-s-1) = f(x)u(0, l-1)$ над модулем M длины $l-s$, определяемый по правилу

$$v(0, l-s-1) = f(x)u(0, l-1) = \sum_{t=0}^s f_t x^t u(0, l-1).$$

Другими словами,

$$v(i) = \sum_{t=0}^s f_t u(i+t), \quad i = 0, 1, \dots, l-s-1.$$

Произведение многочлена степени $s \geq l$ на отрезок длины l не определено.

Будем говорить, что многочлен $f(x) \in R[x]$ степени s аннулирует $u(0, l-1)$, если $s \geq l$ или $s < l$ и $f(x)u(0, l-1) = \bar{0}$, где $\bar{0}$ обозначает нулевой отрезок над модулем M подходящей длины (в данном случае длины $l-s$). Унитарный многочлен, аннулирующий отрезок, называется его характеристическим многочленом. Из приведенных определений следует, что унитарный многочлен порождает отрезок тогда и только тогда, когда он является характеристическим многочленом этого отрезка. В этой терминологии минимальный многочлен отрезка — это характеристический многочлен наименьшей степени. Минимальный многочлен отрезка в общем случае определяется не однозначно, например, если $a, b \in R \setminus \{0\}$, $ab = 0$, то последовательность $u = (b, b, b, \dots)$ над кольцом R имеет минимальные многочлены $x - b$ и $x - a - b$. Представленный ниже алгоритм ищет какой-то один минимальный многочлен заданного отрезка.

Заметим, что ранг отрезка есть наименьшее число $r \in \{0, 1, \dots, l-1\}$ такое, что сдвиг отрезка влево на r шагов линейно выражается через сдвиги этого отрезка на меньшее число шагов, то есть

$$x^r u(0, l-1) = \sum_{t=0}^{r-1} c_t x^t u(0, l-2), \quad c_t \in R, \quad t = 0, 1, \dots, r-1.$$

При этом коэффициенты линейной зависимости, взятые с обратным знаком, являются коэффициентами минимального многочлена отрезка

$$\mu(x) = x^r - \sum_{t=0}^{r-1} c_t x^t.$$

Если такого r не существует, то ранг отрезка равен l и минимальным многочленом является произвольный унитарный многочлен над кольцом R степени l . Наша задача заключается в том, чтобы по данному отрезку $u(0, l-1)$ построить какой-либо минимальный многочлен этого отрезка.

Назовем позицию $k \in \{0, 1, \dots, l-1\}$ ненулевого отрезка $u(0, l-1)$ критической, если $u(k) \neq 0$, причем $k = 0$ или $u(0, k-1) = \bar{0}$. Элемент $u(k)$ будем называть первым ненулевым элементом отрезка $u(0, l-1)$. Критической позицией нулевого отрезка длины l по определению будем называть число l . Если k — критическая позиция отрезка $u(0, l-1)$, то k равно числу нулей в начале этого отрезка.

Элементы $a, b \in R$ называются лево-ассоциированными, если $a = rb$ для некоторого $r \in R^*$ и право-ассоциированными, если $a = bs$ для некоторого $s \in R^*$, где R^* — множество обратимых элементов кольца R . Отношение лево-ассоциированности является отношением эквивалентности и разбивает кольцо R на попарно не пересекающиеся классы лево-ассоциированных элементов:

$$R = \{0\} \cup R_1 \cup \dots \cup R_N, \quad N \geq 1. \quad (2)$$

Выберем в каждом классе представителя $r_n \in R_n$, $n = 1, \dots, N$. Тогда $R_n = R^* r_n$. Один из классов лево-ассоциированных элементов совпадает с R^* . Будем предполагать, что $R_1 = R^*$ и $r_1 = e$.

Обозначим через ${}_R(S)$ подмодуль модуля ${}_R M$, порожденный непустым подмножеством $S \subseteq M$. Если $S = \emptyset$, то по определению будем полагать ${}_R(S) = 0$, то

есть $R(S) = 0$ — нулевой подмодуль. Пусть $d(M)$ — длина модуля M , то есть длина его произвольного композиционного ряда (в силу теоремы Жордана–Гельдера все такие ряды имеют одинаковую длину, см. [5]). В алгоритме будут строиться подмодули $I_0 = 0, I_1, I_2, \dots, I_t$ такие, что

$$I_1 = R(u_1), I_2 = R(u_1, u_2), \dots, I_t = R(u_1, u_2, \dots, u_t),$$

где $u_r \notin I_{r-1}$, $r = 1, \dots, t$. Для нас важно заметить, что число элементов в системе образующих каждого из этих модулей не превосходит $d(M)$. Действительно, из условия $u_r \notin I_{r-1}$ следует, что цепочка подмодулей $0 = I_0 < I_1 < \dots < I_t$ строго возрастает, откуда $t \leq d(M)$.

Если $0 = I_0 < I_1 < \dots < I_{d(M)} = M$ — композиционный ряд модуля M и p — наименьший простой делитель числа $|M|$, то индекс подгруппы I_{r-1} группы I_r удовлетворяет неравенству $[I_r : I_{r-1}] \geq p$. Следовательно, $|I_r| \geq p^r$, откуда $|M| \geq p^{d(M)}$, или $d(M) \leq \log_p |M|$. В частности, $d(M) \leq \log_2 |M|$.

2. Алгоритм Берлекэмп–Месси для отрезка над модулем

Алгоритм нахождения минимального многочлена отрезка $u(0, l-1)$ длины l над модулем ${}_R M$ заключается в последовательном заполнении N таблиц, соответствующих классам $R_1, \dots, R_N$ лево-ассоциированных элементов кольца R , и в последовательном вычислении подмодулей

$$I_s(k) < {}_R M, \quad k = 0, 1, \dots, l-s-1, \quad s = 0, 1, 2, \dots$$

Вначале все N таблиц пустые. На шаге с номером $s = 0$ в n -ю таблицу, $n = 1, \dots, N$, заносятся многочлен $f_{0n}(x) = r_n$ и отрезок $u_{0n}(0, l-1) = r_n u(0, l-1)$ длины l . На s -м шаге в n -ю таблицу, $n = 1, \dots, N$, заносятся некоторый многочлен $f_{sn}(x)$ над кольцом R степени s со старшим коэффициентом r_n и отрезок $u_{sn}(0, l-s-1) = f_{sn}(x)u(0, l-1)$ над модулем M длины $l-s$. Способ вычисления многочленов $f_{sn}(x)$ представлен далее. Алгоритм заканчивает работу, как только в таблице с номером $n = 1$ появится нулевой отрезок $u_{s1}(0, l-s-1) = \bar{0}$, и тогда минимальный многочлен отрезка полагается равным $f_{s1}(x)$. Если же в таблице с номером $n = 1$ за l шагов не появится нулевой отрезок, то алгоритм заканчивает работу на l -м шаге, и минимальный многочлен отрезка $u(0, l-1)$ полагается равным x^l .

Параллельно с заполнением таблиц на каждом шаге $s = 0, 1, 2, \dots$ вычисляются подмодули $I_s(k)$, $k = 0, 1, \dots, l-s-1$, необходимые для построения многочленов $f_{sn}(x)$. Подмодуль $I_s(k)$ порождается всеми элементами $u_{tn}(k)$, стоящими на k -х местах всех отрезков всех N таблиц, полученных на шагах $t \leq s$ алгоритма, и находящимися до первого ненулевого элемента соответствующего отрезка, включая этот ненулевой элемент. Другими словами, если $k_{tn} \in \{0, \dots, l-t\}$ — критическая позиция отрезка $u_{tn}(0, l-t-1)$, то

$$I_s(k) = {}_R(u_{tn}(k) \mid t = 0, 1, \dots, s, n = 1, \dots, N, k \leq k_{tn}).$$

Если таких элементов $u_{tn}(k)$ нет, то $I_s(k) = (\emptyset) = 0$. Подмодуль $I_s(k)$ хранится в виде системы образующих, описанных выше элементов $u_{tn}(k)$, причем вместе с

элементом $u_{tn}(k)$ хранятся числа t, n , показывающие, из какой строки t и из какой таблицы n взят этот элемент.

Приведем теперь полное описание алгоритма.

Задан отрезок $u(0, l-1)$ длины l над модулем ${}_R M$. Требуется найти минимальный многочлен $\mu(x) \in R[x]$ отрезка $u(0, l-1)$.

Алгоритм заключается в последовательном выполнении шагов $s = 0, 1, 2, \dots$. Предварительно представляем кольцо R в виде объединения (2) классов лево-ассоциированных элементов и выбираем в каждом ненулевом классе R_n произвольного представителя $r_n \in R_n, n = 1, \dots, N$. При этом мы считаем, что $R_1 = R^*$ и $r_1 = e$.

Шаг 0. В n -ю таблицу, $n = 1, \dots, N$, заносим многочлен $f_{0n}(x) = r_n$, многочлен степени 0 со старшим коэффициентом r_n , и отрезок

$$u_{0n}(0, l-1) = r_n u(0, l-1) = f_{0n}(x) u(0, l-1)$$

длины l .

Если $u_{01}(0, l-1) = u(0, l-1)$ — нулевой отрезок, то полагаем $\mu(x) = f_{01}(x) = e$, и алгоритм заканчивает работу.

В противном случае пусть $k_{0n} \in \{0, 1, \dots, l\}$ — критическая позиция отрезка $u_{0n}(0, l-1)$, $n = 1, \dots, N$, то есть число нулей в начале этого отрезка. Вычисляем подмодули

$$I_0(k) = (u_{0n}(k) \mid n = 1, \dots, N, k \leq k_{0n}), \quad k = 0, 1, \dots, l-1.$$

При этом вместо подмодуля $I_0(k)$ мы храним систему образующих

$$\{u_{0n}(k) \mid n = 1, \dots, N, k \leq k_{0n}\}.$$

Элемент $u_{0n}(k)$ будем добавлять в систему образующих, только если он не принадлежит подмодулю, порожденному уже добавленными к системе образующих элементами. В частности, мы не включаем нулевые элементы в систему образующих подмодуля. Согласно замечанию в конце первого параграфа, в системе образующих подмодуля $I_0(k)$ тогда будет содержаться не более $d(M)$ элементов.

Шаг $s, s \geq 1$, состоит из двух этапов: заполнения таблиц и вычисления подмодулей $I_s(k)$.

Этап I состоит из заполнения таблиц. Для каждой таблицы $n = 1, \dots, N$ выполняются следующие действия.

1. Берем последний отрезок в n -й таблице, это отрезок

$$u_{s-1,n}(0, l-s) = f_{s-1,n}(x) u(0, l-1),$$

и сдвигаем его влево на один шаг. Получаем отрезок

$$u_{sn}^{(0)}(0, l-s-1) = x u_{s-1,n}(0, l-s) = x f_{s-1,n}(x) u(0, l-1)$$

длины $l-s$ над модулем M . Полагаем

$$f_{sn}^{(0)}(x) = x f_{s-1,n}(x),$$

это многочлен над кольцом R степени s со старшим коэффициентом r_n .

2. Если $u_{sn}^{(0)}(0, l-s-1) = \bar{0}$, то полагаем

$$f_{sn}(x) = f_{sn}^{(0)}(x), \quad u_{sn}(0, l-s-1) = u_{sn}^{(0)}(0, l-s-1), \quad j_{sn} = 0$$

и выполняем следующие действия:

если $n = 1$, то полагаем $\mu(x) = f_{s1}(x)$, и алгоритм заканчивает работу;

если $n > 1$, то переходим к обработке следующей таблицы $n + 1$ при $n < N$ или к этапу II при $n = N$.

Если же $u_{sn}^{(0)}(0, l - s - 1) \neq \bar{0}$, то обозначим через $k_{sn}^{(0)} \in \{0, 1, \dots, l - s - 1\}$ критическую позицию этого отрезка, так что

$$u_{sn}^{(0)}(0, l - s - 1) = (0, \dots, 0, u_{sn}^{(0)}(k_{sn}^{(0)}), \dots), \quad u_{sn}^{(0)}(k_{sn}^{(0)}) \neq 0. \quad (3)$$

3. Возможны два случая. В первом случае

$$u_{sn}^{(0)}(k_{sn}^{(0)}) \notin I_{s-1}(k_{sn}^{(0)}).$$

Тогда полагаем

$$f_{sn}(x) = f_{sn}^{(0)}(x), \quad u_{sn}(0, l - s - 1) = u_{sn}^{(0)}(0, l - s - 1), \quad j_{sn} = 0, \quad k_{sn} = k_{sn}^{(0)}$$

и переходим к обработке следующей таблицы при $n < N$ или к этапу II при $n = N$.

4. Во втором случае

$$u_{sn}^{(0)}(k_{sn}^{(0)}) \in I_{s-1}(k_{sn}^{(0)}).$$

Тогда число нулей в начале отрезка (3) можно увеличить, вычитая из него линейную комбинацию отрезков, полученных на предыдущих шагах алгоритма во всех N таблицах. Действительно, подмодуль $I_{s-1}(k_{sn}^{(0)})$ имеет вид

$$I_{s-1}(k_{sn}^{(0)}) = (u_{s_1 n_1}(k_{sn}^{(0)}), \dots, u_{s_\nu n_\nu}(k_{sn}^{(0)})),$$

где $s_i \in \{0, 1, \dots, s - 1\}$, $n_i \in \{1, \dots, N\}$ и число образующих $\nu = \nu(s - 1, k_{sn}^{(0)})$ подмодуля $I_{s-1}(k_{sn}^{(0)})$ не превосходит $d(M)$. Так как $u_{sn}^{(0)}(k_{sn}^{(0)}) \in I_{s-1}(k_{sn}^{(0)})$, то

$$u_{sn}^{(0)}(k_{sn}^{(0)}) = \sum_{i=1}^{\nu} a_i u_{s_i n_i}(k_{sn}^{(0)})$$

для некоторых элементов $a_1, \dots, a_\nu \in R$, которые можно найти (в худшем случае) перебором. Тогда у отрезка

$$u_{sn}^{(1)}(0, l - s - 1) = u_{sn}^{(0)}(0, l - s - 1) - \sum_{i=1}^{\nu} a_i u_{s_i n_i}(0, l - s_i - 1)$$

на первых $k_{sn}^{(0)} + 1$ местах находятся нули. Положим

$$f_{sn}^{(1)}(x) = f_{sn}^{(0)}(x) - \sum_{i=1}^{\nu} a_i f_{s_i n_i}(x).$$

Тогда

$$u_{sn}^{(1)}(0, l - s - 1) = f_{sn}^{(1)}(x)u(0, l - 1),$$

причем $f_{sn}^{(1)}(x)$ — многочлен степени s со старшим коэффициентом r_n , поскольку степени многочленов $f_{s_i n_i}(x)$ равны $s_i < s$.

5. Для отрезка $u_{sn}^{(1)}(0, l-s-1)$ проделываем те же действия (с очевидными заменами индексов 0 на 1 и 1 на 2), что и для отрезка $u_{sn}^{(0)}(0, l-s-1)$, они изложены в пп. 2, 3, 4 алгоритма. Получим отрезок $u_{sn}^{(2)}(0, l-s-1)$, число нулей в начале которого больше, чем у отрезка $u_{sn}^{(1)}(0, l-s-1)$. И так далее, до тех пор, пока при некотором $j = j_{sn} \in \{0, 1, \dots, l-s-1\}$ не получим отрезок $u_{sn}^{(j)}(0, l-s-1)$, который либо равен $\bar{0}$, либо его первый ненулевой элемент $u_{sn}^{(j)}(k_{sn}^{(j)})$ не принадлежит подмодулю $I_{s-1}(k_{sn}^{(j)})$. В этом случае, как и в пп. 2, 3 мы полагаем

$$f_{sn}(x) = f_{sn}^{(j_{sn})}(x), \quad u_{sn}(0, l-s-1) = u_{sn}^{(j_{sn})}(0, l-s-1), \quad k_{sn} = k_{sn}^{(j_{sn})}.$$

При этом $f_{sn}(x)$ — многочлен степени s со старшим коэффициентом r_n и

$$u_{sn}(0, l-s-1) = f_{sn}(x)u(0, l-1).$$

Затем выполняем следующие действия:

если $n = 1$ и $u_{s1}(0, l-s-1) = \bar{0}$, то полагаем $\mu(x) = f_{s1}(x)$, и алгоритм заканчивает работу;

в остальных случаях переходим к обработке следующей таблицы при $n < N$ или к этапу II при $n = N$.

Этап II состоит в вычислении подмодулей $I_s(k)$.

После того, как в каждую таблицу $n = 1, \dots, N$ добавлен многочлен $f_{sn}(x)$ и отрезок $u_{sn}(0, l-s-1)$, вычисляем подмодули $I_s(k)$, $k = 0, 1, \dots, l-s-1$. Подмодуль $I_s(k)$ порождается подмодулем $I_{s-1}(k)$ и всеми элементами $u_{sn}(k)$, стоящими на k -х местах добавленных в таблицы отрезков до их первого ненулевого элемента, включая сам первый ненулевой элемент. Другими словами, для нахождения системы образующих подмодуля $I_s(k)$ к системе образующих подмодуля $I_{s-1}(k)$ добавляем элементы $u_{sn}(k)$ такие, что $n \in \{1, \dots, N\}$ и $k \leq k_{sn}$. Вместе с элементом $u_{sn}(k)$ хранятся числа s, n , показывающие, из какой строки s и из какой таблицы n взят этот элемент.

Элемент $u_{sn}(k)$ будем добавлять только в том случае, когда он не принадлежит подмодулю, порожденному уже имеющимися в системе образующих элементами. Тогда, согласно замечанию в конце предыдущего параграфа, в системе образующих подмодуля $I_s(k)$ будет содержаться не более $d(M)$ элементов.

Шаг l . Если были выполнены шаги $s = 0, 1, \dots, l-1$ и алгоритм не остановился в п. 2 или 5 ни на каком шаге s , то последний отрезок в каждой таблице имеет длину 1 и сдвиг его влево приводит к пустому отрезку. В этом случае полагаем

$$j_{l1} = 0, \quad f_{l1}(x) = f_{l1}^{(j_{l1})}(x) = \mu(x) = x^l,$$

и алгоритм заканчивает работу.

3. Обоснование алгоритма

Предположим, что алгоритм остановился на r -м шаге, $r \in \{0, 1, \dots, l\}$, и докажем, что $\mu(x) = f_{r1}(x)$ — минимальный многочлен отрезка $u(0, l-1)$.

Лемма 1. Пусть $J_s(k)$, $s \in \{0, 1, \dots, l-1\}$, $k \in \{0, 1, \dots, l-s-1\}$, — подмодуль модуля M , состоящий из всех элементов $v(k)$, где

$$v(0, l-t-1) = g(x)u(0, l-1), \quad (4)$$

$$g(x) \in R[x], \quad \deg g(x) = t \leq s, \quad k = 0 \text{ или } v(0, k-1) = \bar{0}. \quad (5)$$

Тогда

$$I_s(k) = J_s(k), \quad s = 0, 1, \dots, r-1, \quad k = 0, 1, \dots, l-s-1,$$

где $I_s(k)$ — подмодули, построенные в алгоритме.

Доказательство. То, что $J_s(k)$ — подмодуль модуля M , следует из его определения. Очевидно, $I_s(k) \subseteq J_s(k)$, поскольку подмодуль $J_s(k)$ порождается элементами $v(k)$ для всех многочленов $g(x)$, удовлетворяющих условиям (4), а подмодуль $I_s(k)$ — лишь для некоторых многочленов $g(x)$, а именно, для многочленов $f_{tn}(x)$, $t = 0, 1, \dots, s$, $n = 1, \dots, N$, которые вошли в таблицы, заполненные в ходе алгоритма на шагах $0, 1, \dots, s$.

Обратное включение докажем индукцией по s . Проверим, что $J_0(k) \subseteq I_0(k)$. Поскольку многочлен $g(x)$ степени $\deg g(x) \leq s = 0$ есть константа $c \in R$, подмодуль $J_0(k)$ состоит из всех элементов $v(k)$, где

$$v(0, l-1) = cu(0, l-1), \quad c \in R, \quad k = 0 \text{ или } v(0, k-1) = \bar{0}.$$

Если $c = 0$, то $v(k) = 0 \in I_0(k)$. Пусть $c \neq 0$. Тогда c принадлежит некоторому классу лево-ассоциированных элементов R_n , $n \in \{1, \dots, N\}$. Поэтому $c = rr_n$ для некоторого $r \in R^*$. Согласно шагу 0 алгоритма, в n -й таблице находится отрезок

$$r_n u(0, l-1) = r^{-1}cu(0, l-1) = r^{-1}v(0, l-1),$$

и элемент $r^{-1}v(k)$ принадлежит подмодулю $I_0(k)$. Поэтому $v(k) \in I_0(k)$. Отсюда следует, что $J_0(k) \subseteq I_0(k)$.

Проведем шаг индукции. Предположим, что $s \in \{1, \dots, r-1\}$ и что $I_t(k) = J_t(k)$ при всех $t = 0, 1, \dots, s-1$, $k = 0, 1, \dots, l-t-1$. Докажем, что $J_s(k) \subseteq I_s(k)$ при всех $k = 0, 1, \dots, l-s-1$.

Пусть $a \in J_s(k)$, $a \neq 0$. Тогда существует многочлен $g(x) \in R[x]$ степени $t \leq s$ такой, что

$$g(x)u(0, l-1) = v(0, l-t-1) = (0, \dots, 0, a, \dots), \quad (6)$$

где последний отрезок начинается с k нулей. Если $t < s$, то $a \in J_t(k)$ и, в силу предположения индукции, $a \in I_t(k)$. Так как согласно алгоритму

$$I_t(k) \subseteq I_{t+1}(k) \subseteq \dots \subseteq I_s(k),$$

$a \in I_s(k)$ и шаг индукции завершен. Поэтому будем предполагать, что $t = s$. Многочлен $g(x)$ тогда представляется в виде

$$g(x) = cx^s + g_1(x), \quad c \in R \setminus 0, \quad \deg g_1(x) < s.$$

Элемент c принадлежит некоторому классу лево-ассоциированных элементов R_n , $n \in \{1, \dots, N\}$. Поэтому $c = rr_n$ для некоторого $r \in R^*$.

Рассмотрим s -ю строку n -й таблицы, построенной в алгоритме. В ней находится отрезок

$$f_{sn}(x)u(0, l-1) = u_{sn}(0, l-s-1), \quad (7)$$

где

$$f_{sn}(x) = r_n x^s + f_1(x), \quad \deg f_1(x) < s.$$

При этом, согласно алгоритму, отрезок (7) либо нулевой, либо имеет вид

$$u_{sn}(0, l-s-1) = (0, \dots, 0, b, \dots),$$

где последний отрезок начинается с i нулей и

$$i = k_{sn}, \quad b = u_{sn}(i), \quad b \notin I_{s-1}(i), \quad b \in I_s(i). \quad (8)$$

Умножая соотношение (7) слева на r и вычитая из (6), получим, что для многочлена

$$h(x) = g(x) - r f_{sn}(x) = g_1(x) - r f_1(x), \quad \deg h(x) < s,$$

выполняется соотношение

$$h(x)u(0, l-1) = v(0, l-s-1) - r u_{sn}(0, l-s-1).$$

Если $u_{sn}(0, l-s-1) = \bar{0}$, то

$$h(x)u(0, l-1) = v(0, l-s-1) = (0, \dots, 0, a, \dots), \quad \deg h(x) < s,$$

где последний отрезок начинается с k нулей. Тогда $a \in J_{s-1}(k)$. Так как $I_{s-1}(k) = J_{s-1}(k)$ по предположению индукции и $I_{s-1}(k) \subseteq I_s(k)$ в силу алгоритма, то $a \in I_s(k)$, что и требовалось доказать.

Пусть теперь $u_{sn}(0, l-s-1) \neq \bar{0}$. Тогда

$$h(x)u(0, l-1) = (0, \dots, 0, a, \dots) - (0, \dots, 0, rb, \dots),$$

где первый и второй отрезки начинаются с k и i нулей соответственно.

Возможны три случая.

Пусть $k < i$. Тогда

$$h(x)u(0, l-1) = (0, \dots, 0, a, \dots), \quad \deg h(x) < s,$$

где отрезок в правой части начинается с k нулей. Следовательно, $a \in J_{s-1}(k)$. Отсюда $a \in I_{s-1}(k) \subseteq I_s(k)$, что и требовалось доказать.

Пусть $k > i$. Тогда

$$h(x)u(0, l-1) = (0, \dots, 0, -rb, \dots), \quad \deg h(x) < s,$$

где отрезок в правой части начинается с i нулей. Следовательно, $-rb \in J_{s-1}(i)$. В силу предположения индукции $-rb \in I_{s-1}(i)$. Поскольку элемент r обратим, то $b \in I_{s-1}(i)$, что противоречит (8). Таким образом, этот случай невозможен.

Пусть $k = i$. Тогда

$$h(x)u(0, l-1) = (0, \dots, 0, a - rb, \dots), \quad \deg h(x) < s,$$

где отрезок в правой части начинается с $k = i$ нулей. Следовательно, $a - rb \in J_{s-1}(i)$. С учетом предположения индукции $a - rb \in I_{s-1}(i) \subseteq I_s(i)$. Так как $b \in I_s(i)$ в силу (8), $a \in I_s(i) = I_s(k)$, что и требовалось доказать. Лемма доказана.

Теорема 1. *Вырабатываемый алгоритмом многочлен $\mu(x)$ является минимальным многочленом отрезка $u(0, l-1)$.*

Доказательство. Если алгоритм останавливается на шаге $r = 0$, то $u(0, l-1) = \bar{0}$ и, очевидно, вырабатываемый алгоритмом многочлен $\mu(x) = e$ является минимальным многочленом отрезка $u(0, l-1)$.

Пусть алгоритм останавливается на шаге $r \in \{1, \dots, l\}$. Тогда, согласно пп. 2, 5 и шагу l алгоритма, результатом его работы является унитарный многочлен

$$\mu(x) = f_{r1}(x) = f_{r1}^{(j_{r1})}(x)$$

степени r . Если $r < l$, то, согласно пп. 2, 5 алгоритма,

$$f_{r1}(x)u(0, l-1) = u_{r1}^{(j_{r1})}(0, l-r-1) = \bar{0},$$

и, следовательно, $f_{r1}(x)$ — характеристический многочлен отрезка $u(0, l-1)$. Если $r = l$, то, согласно шагу l алгоритма, $f_{r1}(x) = x^l$, и снова $f_{r1}(x)$ — характеристический многочлен отрезка $u(0, l-1)$, так как его степень равна длине отрезка. Остается доказать, что $f_{r1}(x)$ — характеристический многочлен наименьшей степени.

Предположим, что это не так. Тогда существует характеристический многочлен $g(x)$ отрезка $u(0, l-1)$ степени $s < r$. При этом $s \geq 1$, поскольку при $s = 0$ многочлен $g(x) = e$ был бы минимальным многочленом отрезка $u(0, l-1)$, а в этом случае $u(0, l-1)$ — нулевой отрезок и алгоритм остановился бы на шаге $r = 0$. Рассмотрим отрезок, находящийся в s -й строке таблицы с номером $n = 1$,

$$u_{s1}(0, l-s-1) = f_{s1}(x)u(0, l-1).$$

Этот отрезок не является нулевым отрезком, поскольку алгоритм не закончил работу на s -м шаге. Следовательно,

$$u_{s1}(0, l-s-1) = (0, \dots, 0, a, \dots),$$

где последний отрезок начинается с k нулей и, согласно алгоритму,

$$k = k_{s1}, \quad a \notin I_{s-1}(k), \quad a \in I_s(k). \quad (9)$$

Поскольку $g(x)$ и $f_{s1}(x)$ — унитарные многочлены над R степени s , их разность $h(x) = g(x) - f_{s1}(x)$ есть многочлен степени, меньшей s . Ясно, что

$$\begin{aligned} h(x)u(0, l-1) &= g(x)u(0, l-1) - u_{s1}(0, l-s-1) \\ &= \bar{0} - u_{s1}(0, l-s-1) = (0, \dots, 0, -a, \dots), \quad \deg h(x) < s, \end{aligned}$$

где последний отрезок начинается с k нулей. Отсюда, по определению подмодуля $J_{s-1}(k)$, следует, что $a \in J_{s-1}(k)$. Тогда, в силу леммы 1, $a \in I_{s-1}(k)$, что противоречит (9). Следовательно, вырабатываемый алгоритмом многочлен $\mu(x) = f_{r1}(x)$ есть характеристический многочлен отрезка $u(0, l-1)$ наименьшей степени.

Оценим трудоемкость алгоритма. Под операцией будем понимать любую из четырех арифметических операций в кольце R , операцию сложения элементов модуля M и умножения элемента кольца R на элемент модуля M , операцию сравнения двух

элементов модуля M , а также арифметическую операцию или операцию сравнения в кольце целых чисел. Мы не учитываем операции обращения в память, поскольку их число может быть существенно оптимизировано при конкретной реализации алгоритма по сравнению с тем, как это изложено в данной статье. Так же, как и для арифметических операций, число обращений в память составляет величину $O(l^2)$, но при подсчете константы при l^2 операции обращения в память учитываться не будут.

Теорема 2. *Трудоёмкость алгоритма оценивается величиной $O(l^2)$ при $l \rightarrow \infty$. Константа при l^2 (без учета операций обращения в память) не превосходит величины $4Nd(M) + 2N$, где N — число классов лево-ассоциированных ненулевых элементов кольца R , $d(M)$ — длина модуля ${}_R M$. При этом $N < |R|$ и $d(M) \leq \log_2 |M|$.*

Доказательство. Вначале нужно представить кольцо R в виде объединения (2) классов попарно лево-ассоциированных элементов. Это заведомо можно сделать за $|R|^2$ операций (составление таблицы Кели). Поэтому трудоёмкость данного этапа вообще не зависит от l , то есть равна $O(1)$. В дальнейшем действия алгоритма, имеющие трудоёмкость $O(1)$, не учитываются.

Трудоёмкость шага 0 алгоритма включает $2Nl$ операций на заполнение таблиц и не более $|R|^{d(M)} 2d(M)N = O(1)$ операций на составление систем образующих подмодулей $I_0(k)$. Здесь $|R|^{d(M)} 2d(M)$ — оценка трудоёмкости проверки принадлежности элемента $u \in M$ подмодулю $I < M$, порожденному не более $d(M)$ элементами. Множитель N возникает потому, что в каждой из N таблиц имеется не более одного элемента $u_{0n}(k_{0n}) \neq 0$, который может быть добавлен в систему образующих соответствующего подмодуля $I_0(k_{0n})$.

Пусть алгоритм остановился на r -м шаге, $r \in \{0, 1, \dots, l\}$. Оценим трудоёмкость шагов $s = 1, \dots, r$. Вначале для фиксированной таблицы $n \in \{1, \dots, N\}$ оценим трудоёмкость выполнения этапов I при всех $s = 1, \dots, r$.

Трудоёмкость п. 1 и п. 2 алгоритма есть $O(1)$ и l соответственно. В пп. 3, 4 требуется устанавливать принадлежность элемента $u \in M$ подмодулю $I < M$, заданному своей системой образующих $I = {}_R(u_1, \dots, u_\nu)$, и в случае $u \in I$ представлять элемент u в виде $u = a_1 u_1 + \dots + a_\nu u_\nu$, где $a_1, \dots, a_\nu \in R$. Обе эти задачи можно решить перебором всех $|R|^\nu$ сумм $a_1 u_1 + \dots + a_\nu u_\nu$, где $a_1, \dots, a_\nu \in R$. Поскольку вычисление каждой такой суммы и сравнение ее с элементом u занимает 2ν операций и, как отмечалось в конце параграфа 1, число ν образующих подмодуля $I < M$ не превосходит $d(M)$, данная задача решается не более, чем за $|R|^\nu 2\nu \leq |R|^{d(M)} 2d(M)$ операций. Теперь легко подсчитать, что трудоёмкость пп. 3 и 4 алгоритма не превосходит $|R|^{d(M)} 2d(M) + 4d(M)l$ операций.

Согласно п. 5, для каждого шага s алгоритма п. 2 выполняется $j_{sn} + 1$ раз, а пп. 3, 4 выполняются j_{sn} раз. Суммируя трудоёмкости пп. 1–5, получим, что трудоёмкость обработки n -ой таблицы на этапе I шага s оценивается величиной

$$l(j_{sn} + 1) + (|R|^{d(M)} 2d(M) + 4d(M)l)j_{sn}.$$

Поэтому трудоёмкость обработки n -й таблицы на этапах I всех шагов $s = 1, \dots, r$ не превосходит

$$rl + (|R|^{d(M)} 2d(M) + (4d(M) + 1)l) \sum_{s=1}^r j_{sn}. \quad (10)$$

Для оценки полученной суммы заметим, что если на шаге $s-1$ алгоритма в начале отрезка $u_{s-1,n}(0, l-s)$ было $k_{s-1,n} \geq 0$ нулей, то в начале отрезка $u_{sn}^{(0)}(0, l-s-1)$ находится не менее $k_{s-1,n} - 1$ нулей, и после каждого выполнения пп. 3, 4 на s -м шаге число нулей в начале отрезка увеличивается не менее, чем на 1. Следовательно, после j_{sn} повторений пп. 3, 4 будет получен отрезок

$$u_{sn}^{(j_{sn})}(0, l-s-1) = u_{sn}(0, l-s-1),$$

в начале которого находится $k_{sn} \geq k_{s-1,n} - 1 + j_{sn}$ нулей. Отсюда получаем цепочку неравенств

$$\begin{aligned} k_{sn} &\geq k_{s-1,n} - 1 + j_{sn} \geq k_{s-2,n} - 2 + j_{s-1,n} + j_{sn} \\ &\geq \dots \geq k_{0n} - s + j_{1n} + \dots + j_{s-1,n} + j_{sn}. \end{aligned}$$

Так как число нулей в начале отрезка $u_{sn}(0, l-s-1)$ не превосходит его длины,

$$l-s \geq k_{sn} \geq k_{0n} - s + j_{1n} + \dots + j_{s-1,n} + j_{sn}.$$

Отсюда,

$$j_{1n} + \dots + j_{s-1,n} + j_{sn} \leq l - k_{0n} \leq l.$$

Полагая $s = r$ и подставляя в (10), получаем, что трудоемкость обработки n -й таблицы на этапах I всех шагов $s = 1, \dots, r$ оценивается сверху величиной

$$rl + (|R|^{d(M)} 2d(M) + (4d(M) + 1)l)l.$$

Умножая на число таблиц N , получаем, что трудоемкость этапов I на всех шагах $s = 1, \dots, r$ алгоритма не превосходит

$$Nrl + (|R|^{d(M)} 2d(M) + (4d(M) + 1)l)Nl.$$

Аналогично тому, как объяснялось при подсчете трудоемкости шага 0, находим, что трудоемкость этапа II на шаге s не превосходит $|R|^{d(M)} 2d(M)N$ операций. Следовательно, трудоемкость этапов II на шагах $s = 1, \dots, r$ алгоритма составляет $|R|^{d(M)} 2d(M)Nr$ операций.

Трудоемкость шага l составляет $O(1)$.

Складывая трудоемкости шага 0, этапов I и этапов II на шагах $s = 1, \dots, r$, и учитывая, что $r \leq l$, получим, что трудоемкость алгоритма (с точностью до $O(1)$) не превосходит

$$\begin{aligned} 2Nl + Nrl + (|R|^{d(M)} 2d(M) + (4d(M) + 1)l)Nl + |R|^{d(M)} 2d(M)Nr \\ \leq (4Nd(M) + 2N)l^2 + (4|R|^{d(M)}Nd(M) + 2N)l \end{aligned}$$

операций, что и требовалось доказать.

4. Алгоритм для отрезков над кольцами и бимодулями

Прежде всего отметим, что алгоритм из параграфа 3 можно аналогично изложить для конечного правого модуля M_R над конечным кольцом R с единицей. Основное

отличие заключается в том, что вместо разложения (2) нужно брать разложение кольца R в объединение классов право-ассоциированных элементов. Кроме того, коэффициенты кольца R нужно писать справа от элементов модуля M и, например, закон рекурсии (1) для отрезка $u(0, l-1)$ над правым модулем M_R будет иметь вид

$$u(i+s) = u(i+s-1)c_{s-1} + \dots + u(i+1)c_1 + u(i)c_0, \quad i = 0, 1, \dots, l-s-1. \quad (11)$$

Многочлены $g(x) \in R[x]$ умножаются на отрезки справа. В частности, унитарный многочлен $g(x)$ степени s порождает отрезок $u(0, l-1)$ тогда и только тогда, когда $s \geq l$ или $s < l$ и $u(0, l-1)g(x) = \bar{0}$. Подмодуль правого модуля M_R , порожденный подмножеством $S \subseteq M$, обозначается $(S)_R$, то есть кольцо R пишется справа.

Рассмотрим случай отрезка над конечным бимодулем.

Пусть ${}_A M_B$ — конечный бимодуль над конечными кольцами A и B с единицами [5]. Говорим, что ненулевой отрезок $u(0, l-1) \in M^l$ имеет ранг $r \in \{1, \dots, l-1\}$, если существуют константы

$$k \in \mathbb{N}, \quad a_1, \dots, a_k \in A, \quad b_1, \dots, b_k \in B, \quad i_1, \dots, i_k \in \{0, 1, \dots, r-1\}$$

такие, что

$$u(i+r) = \sum_{t=1}^k a_t u(i+i_t) b_t, \quad i = 0, 1, \dots, l-r-1, \quad (12)$$

и r является наименьшим числом из интервала $\{1, \dots, l-1\}$ с этим свойством. Если такого r не существует, то будем говорить, что ранг отрезка $u(0, l-1)$ равен l . Ранг нулевого отрезка по определению равен 0. Соотношение (12) будем называть двусторонним законом рекурсии для отрезка $u(0, l-1)$. Наша цель заключается в нахождении ранга заданного отрезка $u(0, l-1)$ над бимодулем ${}_A M_B$ и какого-либо двустороннего закона рекурсии (12). Покажем, как эта задача может быть сведена к случаю отрезка над левым модулем ${}_R M$.

Пусть $(B^{op}, +, *)$ — кольцо, противоположное (антиизоморфное) кольцу B . Умножение в кольце B^{op} задается равенством $b_1 * b_2 = b_2 b_1$. Обозначим через

$$R = A \otimes B^{op} = A \otimes_{\mathbb{Z}} B^{op}$$

тензорное произведение колец A и B^{op} над $\mathbb{Z}$ (см. [5]). Элементами кольца R являются формальные суммы

$$\sum_{t=1}^k a_t \otimes b_t, \quad k \in \mathbb{N}, a_t \in A, b_t \in B^{op}.$$

Определим операцию внешнего умножения элементов кольца R на элементы из M , полагая $(a \otimes b)m = amb$. Тогда абелева группа $(M, +)$ превращается в левый R -модуль. Легко проверить, что подмножество $I \subseteq M$ является подбимодулем в ${}_A M_B$ тогда и только тогда, когда I — подмодуль модуля ${}_R M$. Кроме того, подбимодуль ${}_A(S)_B$ бимодуля ${}_A M_B$, порожденный подмножеством $S \subseteq M$ совпадает с подмодулем ${}_R(S)$ модуля ${}_R M$, порожденным S .

Группируя слагаемые в (12), получим соотношение

$$u(i+r) = c_{r-1}u(i+r-1) + \dots + c_1u(i+1) + c_0u(i), \quad i = 0, 1, \dots, l-r-1,$$

где

$$c_s = \sum_{1 \leq t \leq k, i_t = s} a_t \otimes b_t, \quad s = 0, 1, \dots, r-1,$$

— элементы кольца R . Отсюда следует, что ранг отрезка $u(0, l-1)$ над бимодулем ${}_A M_B$ равен рангу этого отрезка над левым модулем ${}_R M$, где $R = A \otimes B^{op}$. При этом нахождение двустороннего закона рекурсии (12) эквивалентно нахождению минимального многочлена отрезка над модулем ${}_R M$. Поскольку кольцо $A \otimes B^{op}$ конечно, поставленная задача сводится к применению алгоритма из параграфа 2 к отрезку $u(0, l-1)$ над модулем ${}_R M$. Трудоемкость алгоритма оценивается сверху величиной

$$(4Nd(M) + 2N)l^2 + O(l),$$

где N — число классов лево-ассоциированных элементов кольца $R = A \otimes B^{op}$ и $d(M)$ — длина модуля ${}_R M$, совпадающая с длиной бимодуля ${}_A M_B$.

Рассмотрим теперь случай отрезка над конечным кольцом R . Этот случай является частным случаем для всех рассмотренных выше при $M = R$. Его особенность заключается в том, что кольцо R можно рассматривать как левый модуль ${}_R R$, как правый модуль R_R и как бимодуль ${}_R R_R$. Соответственно будут определены три различных ранга отрезка $u(0, l-1)$ над кольцом R , которые можно называть левым, правым и двусторонним рангами. Аналогично имеются три вида законов рекурсии для отрезка $u(0, l-1)$, левый, правый и двусторонний, а также три минимальных многочлена этого отрезка, левый и правый минимальные многочлены над кольцом R и двусторонний минимальный многочлен над кольцом $R \otimes R^{op}$ (или двусторонний закон рекурсии (12) над кольцом R). Все они могут быть найдены с помощью алгоритма, изложенного в параграфе 2, при $M = R$. Отметим, что в общем случае левый и правый ранги отрезка над кольцом R не равны друг другу, а двусторонний ранг, очевидно, не превосходит каждого из них.

При нахождении левого ранга и левого минимального многочлена отрезка над кольцом R применяется алгоритм, изложенный в параграфе 2, для модуля ${}_R R$. При этом подмодули $I_s(k)$, вычисляемые в алгоритме, являются подмодулями модуля ${}_R R$, то есть левыми идеалами кольца R . При нахождении правого ранга и правого минимального многочлена подмодули $I_s(k)$ являются подмодулями модуля R_R , то есть правыми идеалами кольца R . При нахождении двустороннего ранга и двустороннего закона рекурсии отрезка над кольцом R подмодули $I_s(k)$ являются подмодулями модуля ${}_S R$, где $S = R \otimes R^{op}$, то есть, как легко видеть, двусторонними идеалами кольца R .

5. Пример

Пусть $P_{m,m}$ — кольцо квадратных $m \times m$ матриц над конечным полем $P = GF(q)$. Кольцо $P_{m,m}$ является некоммутативным кольцом порядка q^{m^2} . Две матрицы над полем являются лево-ассоциированными тогда и только тогда, когда они строчно эквивалентны, то есть одна из них переводится в другую эквивалентными преобразованиями строк. Это равносильно тому, что эти матрицы строчно эквивалентны одной и той же специальной ступенчатой матрице. Под специальной ступенчатой матрицей понимается ступенчатая матрица, у которой первый ненулевой элемент в каждой ненулевой строке равен единице, все элементы над ним равны 0,

и если некоторая строка матрицы нулевая, то и все последующие строки нулевые. Специальные ступенчатые матрицы являются представителями классов лево-ассоциированных элементов кольца $P_{m,m}$ квадратных матриц, и число N классов ненулевых лево-ассоциированных элементов этого кольца равно числу ненулевых специальных ступенчатых матриц. Например, для матриц размеров 2×2 все ненулевые специальные ступенчатые матрицы имеют вид

$$\begin{aligned} A_1 = E &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \\ \{A_2, \dots, A_{q+1}\} &= \left\{ \begin{pmatrix} 1 & a \\ 0 & 0 \end{pmatrix} \middle| a \in P \right\}, \\ A_{q+2} &= \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}. \end{aligned}$$

В частности, $N = q + 2$.

Рассмотрим теперь кольцо $R = P_{2,2}$ квадратных 2×2 матриц над полем $P = GF(3)$. В этом случае $N = 5$, и разбиение кольца R на классы лево-ассоциированных элементов имеет вид

$$\begin{aligned} R &= \{0\} \cup R_1 \cup R_2 \cup R_3 \cup R_4 \cup R_5, \\ |R| &= 81, \quad |R_1| = |R^*| = 48, \quad |R_2| = |R_3| = |R_4| = |R_5| = 8, \end{aligned}$$

а представители $A_n \in R_n$ этих классов равны

$$A_1 = E, \quad A_2 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad A_3 = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}, \quad A_4 = \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix}, \quad A_5 = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

Множества R_n легко описать. Действительно, $R_1 = R^*$, R_2 (R_5) — множество ненулевых матриц с нулевым вторым (первым) столбцом, R_3 (R_4) — множество ненулевых матриц, у которых второй столбец равен первому (второй столбец равен первому, умноженному на 2).

В качестве левого модуля ${}_R M$ можно взять, например, множество вектор-столбцов над R длины 2, само кольцо R матриц размера 2×2 , множество матриц над R размера $2 \times k$ для произвольного $k \in \mathbb{N}$. С увеличением k пример становился бы более громоздким. Мы возьмем $k = 2$ и, таким образом, в качестве модуля M рассмотрим само кольцо R . Из примера будет видно, что вычисления с матрицами размеров $2 \times k$, $k \in \mathbb{N}$, проводились бы аналогично.

Рассмотрим следующий отрезок $u = u(0, 7)$ длины 8 над кольцом R :

$$\begin{aligned} u &= u(0, 7) \\ &= \left(\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right). \end{aligned}$$

и найдем левый минимальный многочлен этого отрезка над кольцом R . Для этого будем заполнять $N = 5$ таблиц, соответствующих классам R_n , $n = 1, \dots, 5$, ненулевых лево-ассоциированных элементов кольца R . В отдельной таблице вычисляются левые идеалы $I_s(k)$.

На шаге $s = 0$ в n -ю таблицу, $n = 1, \dots, 5$, заносятся многочлен $f_{0n}(x) = A_n$ и отрезок $u_{0n}(0, 7) = A_n u(0, 7)$ длины 8. На s -м шаге, $s \geq 1$, в n -й таблице производятся следующие операции. Последний отрезок в этой таблице, то есть отрезок

$u_{s-1,n}(0, 8-s)$, сдвигается влево на один шаг. Получается отрезок $u_{sn}^{(0)}(0, 7-s) = xu_{s-1,n}(0, 8-s)$. Аналогично получается многочлен $f_{sn}^{(0)}(x) = xf_{s-1,n}(x)$. Вслед за этим мы обнуляем как можно больше элементов, стоящих в начале отрезка $u_{sn}^{(0)}(0, 7-s)$, прибавляя к нему (левые) линейные комбинации отрезков, полученных на предыдущих шагах $0, 1, \dots, s-1$ алгоритма во всех 5 таблицах. Эти действия комментируются в последнем столбце таблиц. Соответственно к многочлену $f_{sn}^{(0)}(x)$ прибавляются (левые) линейные комбинации многочленов, полученных на предыдущих шагах алгоритма (эти действия не комментируются, поскольку они такие же, как для отрезков). В результате последовательно получаем отрезки $u_{sn}^{(j)}(0, 7-s)$ и многочлены $f_{sn}^{(j)}(x)$, $j = 0, 1, \dots$. Возможность аннулирования первого ненулевого элемента $u_{sn}^{(j)}(k)$ отрезка $u_{sn}^{(j)}(0, 7-s)$ проверяется с помощью таблицы, содержащей левые идеалы $I_{s-1}(k)$. Если $u_{sn}^{(j)}(k) \in I_{s-1}(k)$, то мы аннулируем элемент $u_{sn}^{(j)}(k)$ и получаем отрезок $u_{sn}^{(j+1)}(0, 7-s)$ и многочлен $f_{sn}^{(j+1)}(x)$. Если же $u_{sn}^{(j)}(k) \notin I_{s-1}(k)$, то переходим к обработке следующей таблицы, а элемент $u_{sn}^{(j)}(k)$ впоследствии включается в систему образующих левого идеала $I_s(k)$.

После обработки всех 5 таблиц на шаге $s \geq 0$ заполняем таблицу с левыми идеалами $I_s(k)$, $k = 0, 1, \dots, 7-s$. Левый идеал $I_s(k)$ получается добавлением к системе образующих левого идеала $I_{s-1}(k)$ (или нулевого идеала при $s = 0$) элементов, стоящих на k -х местах только что полученных 5 последних отрезков в 5 таблицах, причем эти элементы должны быть первыми ненулевыми элементами своих отрезков.

Из таблиц видно, что алгоритм полностью выполнил шаги $s = 0, 1, 2$ и остановился на шаге $s = 3$ при обработке первой таблицы, поскольку в первой таблице был получен нулевой отрезок. Найденный алгоритмом левый минимальный многочлен отрезка равен

$$\mu_l(x) = x^3 + \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix}.$$

Аналогично находится правый минимальный многочлен отрезка $u(0, 7)$ над кольцом R . Алгоритм при этом останавливается на шаге $s = 4$ и вырабатывает правый минимальный многочлен

$$\mu_r(x) = x^4 + \begin{pmatrix} 0 & 0 \\ 0 & 2 \end{pmatrix} x^3 + \begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix} x^2 + \begin{pmatrix} 2 & 0 \\ 1 & 0 \end{pmatrix} x + \begin{pmatrix} 0 & 2 \\ 1 & 0 \end{pmatrix}.$$

Для нахождения двустороннего ранга и двустороннего закона рекурсии отрезка $u(0, 7)$ требуется искать левый ранг и левый минимальный многочлен этого отрезка над левым модулем ${}_S R$, где $S = R \otimes R^{op}$. Тензорное произведение колец матриц размеров $m \times m$ и $n \times n$ изоморфно кольцу матриц размеров $mn \times mn$, а $R_{m,m}^{op} \cong P_{m,m}$. Значит, кольцо $P_{2,2} \otimes P_{2,2}^{op}$ изоморфно кольцу $P_{4,4}$ матриц размеров 4×4 над полем $P = GF(q)$. Несложно подсчитать, что число N классов ненулевых лево-ассоциированных элементов кольца $P_{4,4}$ равно

$$q^4 + 3q^3 + 4q^2 + 3q + 4.$$

При $q = 3$ получаем $N = 211$. Поэтому при реализации алгоритма в нашем примере потребуется заполнять 211 таблиц. Кроме того, необходимо разбивать кольцо S (а не кольцо $P_{4,4}$) на классы лево-ассоциированных элементов. Для этого достаточно

Таблица 1. Таблица для $n = 1$

s	$f_{s1}(x)$ и $f_{s1}^{(j)}(x)$	$u_{s1}(0, l-s-1)$ и $u_{s1}^{(j)}(0, l-s-1)$
0	$f_{01}(x) = E$	$u_{01}(0, 7) = \left(\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right)$
1	$xf_{01}(x) = x$	$xu_{01}(0, 6) = \left(\begin{pmatrix} 0 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right)$ прибавляем $\begin{pmatrix} 0 & 0 \\ 2 & 0 \end{pmatrix} u_{01}(0, 7)$
	$f_{11}(x) = x + \begin{pmatrix} 0 & 0 \\ 2 & 0 \end{pmatrix}$	$u_{11}(0, 6) = \left(\begin{matrix} 0 & \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 2 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \end{matrix} \right)$
2	$f_{21}(x) = x^2 + \begin{pmatrix} 0 & 0 \\ 2 & 0 \end{pmatrix} x$	$u_{21}(0, 5) = \left(\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 2 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \right)$
3	$xf_{21}(x) = x^3 + \begin{pmatrix} 0 & 0 \\ 2 & 0 \end{pmatrix} x^2$	$xu_{21}(0, 5) = \left(\begin{pmatrix} 2 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 2 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \right)$ прибавляем $\begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix} u_{01}(0, 7) + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} u_{21}(0, 5)$
	$f_{31}^{(1)}(x) = x^3 + \begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix}$	$u_{31}^{(1)}(0, 4) = \left(\begin{matrix} 0 & \begin{pmatrix} 1 & 0 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 2 & 1 \end{pmatrix} \end{matrix} \right)$ прибавляем $\begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} u_{11}(0, 6) + \begin{pmatrix} 0 & 0 \\ 2 & 0 \end{pmatrix} u_{04}(0, 7)$
	$f_{31}^{(2)}(x) = x^3 + \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$	$u_{31}^{(2)}(0, 4) = \left(\begin{matrix} 0 & 0 & \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \end{matrix} \right)$ прибавляем $\begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} u_{15}(0, 6)$
	$f_{31}^{(3)}(x) = x^3 + \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 2 & 0 \\ 1 & 1 \end{pmatrix} = f_{31}(x) = \mu_l(x)$	$u_{31}^{(3)}(0, 4) = \left(\begin{matrix} 0 & 0 & 0 & 0 & 0 \end{matrix} \right)$

Таблица 2. Таблица для $n = 2$

s	$f_{s1}(x)$ и $f_{s1}^{(j)}(x)$	$u_{s1}(0, l-s-1)$ и $u_{s1}^{(j)}(0, l-s-1)$
0	$f_{02}(x) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$	$u_{02}(0, 7) = \left(\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \quad 0 \quad \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \right)$
1	$f_{12}(x) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} x$	$u_{12}(0, 6) = \left(\quad 0 \quad \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \right)$
2	$f_{22}(x) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} x^2$	$u_{22}(0, 5) = \left(\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \right)$

Таблица 3. Таблица для $n = 3$

s	$f_{s1}(x)$ и $f_{s1}^{(j)}(x)$	$u_{s1}(0, l-s-1)$ и $u_{s1}^{(j)}(0, l-s-1)$
0	$f_{03}(x) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$	$u_{03}(0, 7) = \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} 0 \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix} 0 \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$
0	$xf_{03}(x) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} x$	$xu_{03}(0, 7) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} 0 \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix} 0 \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$ прибавляем $\begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} u_{01}(0, 7)$
	$f_{13}(x) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}$	$u_{13}(0, 6) = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}$
2	$f_{23}(x) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} x^2 + \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} x$	$u_{23}(0, 5) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}$

Таблица 4. Таблица для $n = 4$

s	$f_{s1}(x)$ и $f_{s1}^{(j)}(x)$	$u_{s1}(0, l-s-1)$ и $u_{s1}^{(j)}(0, l-s-1)$
0	$f_{04}(x) = \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix}$	$u_{04}(0, 7) = \begin{pmatrix} 0 & \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \end{pmatrix}$
0	$xf_{04}(x) = \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} x$	$xu_{04}(0, 7) = \begin{pmatrix} \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} \end{pmatrix}$ прибавляем $\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} u_{01}(0, 7)$
	$f_{14}(x) = \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$	$u_{14}(0, 6) = \begin{pmatrix} 0 & \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} 0 \end{pmatrix}$
2	$f_{24}(x) = \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix} x^2 + \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} x$	$u_{24}(0, 6) = \begin{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} 0 \end{pmatrix}$

Таблица 5. Таблица для $n = 5$

s	$f_{s1}(x)$ и $f_{s1}^{(j)}(x)$	$u_{s1}(0, l-s-1)$ и $u_{s1}^{(j)}(0, l-s-1)$
0	$f_{05}(x) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$	$u_{05}(0, 7) = \left(\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \right)$
1	$xf_{05}(x) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x$	$xu_{05}(0, 7) = \left(\begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \right)$ прибавляем $\begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} u_{01}(0, 7)$
	$f_{15}(x) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix}$	$u_{15}(0, 6) = \left(\begin{matrix} 0 & 0 & \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \end{matrix} \right)$
2	$xf_{15}(x) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x^2 + \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} x$	$xu_{15}(0, 6) = \left(\begin{matrix} 0 & \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \end{matrix} \right)$ прибавляем $u_{04}(0, 7) + u_{11}(0, 6)$
	$f_{25}^{(1)}(x) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x^2 + \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} x + \begin{pmatrix} 1 & 2 \\ 2 & 0 \end{pmatrix}$	$u_{25}^{(1)}(0, 5) = \left(\begin{matrix} 0 & 0 & \begin{pmatrix} 0 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 2 & 0 \end{pmatrix} \end{matrix} \right)$ прибавляем $\begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix} u_{15}(0, 6)$
	$f_{25}^{(2)}(x) = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x^2 + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x + \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix}$ $= f_{25}(x)$	$u_{25}^{(2)}(0, 5) = \left(\begin{matrix} 0 & 0 & 0 & \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix} \end{matrix} \right)$ $= u_{25}(0, 5)$

Таблица 6. Подмодули $I_s(k)$

s	$I_s(0)$	$I_s(1)$	$I_s(2)$	$I_s(3)$	$I_s(4)$	$I_s(5)$	$I_s(6)$	$I_s(7)$
0	$R(u_{01}(0)) = R\left(\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}\right)$	$R(u_{04}(1)) = R\left(\begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix}\right)$	0	0	0	0	0	0
1	$R(u_{01}(0)) = R\left(\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}\right)$	$R(u_{04}(1), u_{11}(1))$ $= R\left(\begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\right) = R$	$R(u_{15}(2))$ $= R\left(\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}\right)$	0	0	0	0	0
2	$R(u_{01}(0), u_{21}(0))$ $= R\left(\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\right) = R$	$R(u_{04}(1), u_{11}(1))$ $= R\left(\begin{pmatrix} 2 & 2 \\ 0 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\right) = R$	$R(u_{15}(2))$ $= R\left(\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}\right)$	$R(u_{25}(3)) =$ $= R\left(\begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}\right)$	0	0	0	0

построить изоморфизм, обратный к изоморфизму $\varphi: S = P_{2,2} \otimes P_{2,2}^{op} \rightarrow P_{4,4}$, что, однако, представляет собой отдельную трудную задачу.

Мы найдем двусторонний закон рекурсии отрезка $u(0, 7)$, не заполняя все $N = 211$ таблиц, а выписывая лишь часть соотношений, аналогичных содержимому этих таблиц. При этом будем руководствоваться целью с помощью отрезка $\mathbf{u} = u(0, 7)$, его сдвигов и их линейных комбинаций получить отрезки, в которых первые ненулевые элементы находятся на местах $k = 0, 1, 2, \dots$. Удастся получить следующие соотношения.

$$\mathbf{u} = u(0, 7) \quad \begin{pmatrix} 11 \\ 11 \end{pmatrix} \begin{pmatrix} 00 \\ 11 \end{pmatrix} \begin{pmatrix} 10 \\ 00 \end{pmatrix} \begin{pmatrix} 22 \\ 11 \end{pmatrix} \begin{pmatrix} 10 \\ 22 \end{pmatrix} \begin{pmatrix} 10 \\ 20 \end{pmatrix} \begin{pmatrix} 21 \\ 00 \end{pmatrix} \begin{pmatrix} 10 \\ 01 \end{pmatrix} \quad (1)$$

$$\begin{pmatrix} 21 \\ 00 \end{pmatrix} \mathbf{u} \quad \begin{pmatrix} 00 \\ 00 \end{pmatrix} \begin{pmatrix} 11 \\ 00 \end{pmatrix} \begin{pmatrix} 20 \\ 00 \end{pmatrix} \begin{pmatrix} 22 \\ 00 \end{pmatrix} \begin{pmatrix} 12 \\ 00 \end{pmatrix} \begin{pmatrix} 10 \\ 00 \end{pmatrix} \begin{pmatrix} 12 \\ 00 \end{pmatrix} \begin{pmatrix} 21 \\ 00 \end{pmatrix} \quad (2)$$

$$x\mathbf{u} \quad \begin{pmatrix} 00 \\ 11 \end{pmatrix} \begin{pmatrix} 10 \\ 00 \end{pmatrix} \begin{pmatrix} 22 \\ 11 \end{pmatrix} \begin{pmatrix} 10 \\ 22 \end{pmatrix} \begin{pmatrix} 10 \\ 20 \end{pmatrix} \begin{pmatrix} 21 \\ 00 \end{pmatrix} \begin{pmatrix} 10 \\ 01 \end{pmatrix} \quad (3)$$

$$x\mathbf{u} + \begin{pmatrix} 00 \\ 20 \end{pmatrix} \mathbf{u} \quad 0 \quad \begin{pmatrix} 10 \\ 00 \end{pmatrix} \begin{pmatrix} 22 \\ 01 \end{pmatrix} \begin{pmatrix} 10 \\ 10 \end{pmatrix} \begin{pmatrix} 10 \\ 10 \end{pmatrix} \begin{pmatrix} 21 \\ 10 \end{pmatrix} \begin{pmatrix} 10 \\ 10 \end{pmatrix} \quad (4)$$

$$x\mathbf{u} + \begin{pmatrix} 00 \\ 20 \end{pmatrix} \mathbf{u} + \begin{pmatrix} 21 \\ 00 \end{pmatrix} \mathbf{u} \begin{pmatrix} 20 \\ 00 \end{pmatrix} \quad 0 \quad 0 \quad \begin{pmatrix} 02 \\ 01 \end{pmatrix} \begin{pmatrix} 20 \\ 00 \end{pmatrix} \begin{pmatrix} 00 \\ 10 \end{pmatrix} \begin{pmatrix} 11 \\ 20 \end{pmatrix} \begin{pmatrix} 00 \\ 10 \end{pmatrix} \quad (5)$$

$$\begin{pmatrix} 11 \\ 01 \end{pmatrix} (5) = \begin{pmatrix} 11 \\ 01 \end{pmatrix} x\mathbf{u} + \begin{pmatrix} 20 \\ 20 \end{pmatrix} \mathbf{u} + \begin{pmatrix} 21 \\ 00 \end{pmatrix} \mathbf{u} \begin{pmatrix} 20 \\ 00 \end{pmatrix} \quad 0 \quad 0 \quad \begin{pmatrix} 00 \\ 01 \end{pmatrix} \begin{pmatrix} 20 \\ 00 \end{pmatrix} \begin{pmatrix} 10 \\ 10 \end{pmatrix} \begin{pmatrix} 01 \\ 20 \end{pmatrix} \begin{pmatrix} 10 \\ 10 \end{pmatrix} \quad (6)$$

$$\begin{pmatrix} 10 \\ 00 \end{pmatrix} (6) = \begin{pmatrix} 11 \\ 00 \end{pmatrix} x\mathbf{u} + \begin{pmatrix} 20 \\ 00 \end{pmatrix} \mathbf{u} + \begin{pmatrix} 21 \\ 00 \end{pmatrix} \mathbf{u} \begin{pmatrix} 20 \\ 00 \end{pmatrix} \quad 0 \quad 0 \quad 0 \quad \begin{pmatrix} 20 \\ 00 \end{pmatrix} \begin{pmatrix} 10 \\ 00 \end{pmatrix} \begin{pmatrix} 01 \\ 00 \end{pmatrix} \begin{pmatrix} 10 \\ 00 \end{pmatrix} \quad (7)$$

$$\begin{pmatrix} 01 \\ 00 \end{pmatrix} (6) \begin{pmatrix} 10 \\ 00 \end{pmatrix} = \begin{pmatrix} 01 \\ 00 \end{pmatrix} x\mathbf{u} \begin{pmatrix} 10 \\ 00 \end{pmatrix} + \begin{pmatrix} 20 \\ 00 \end{pmatrix} \mathbf{u} + \begin{pmatrix} 10 \\ 00 \end{pmatrix} \quad 0 \quad 0 \quad 0 \quad 0 \quad \begin{pmatrix} 10 \\ 00 \end{pmatrix} \begin{pmatrix} 20 \\ 00 \end{pmatrix} \begin{pmatrix} 10 \\ 00 \end{pmatrix} \quad (8)$$

$$\begin{pmatrix} 10 \\ 00 \end{pmatrix} (6) \begin{pmatrix} 00 \\ 10 \end{pmatrix} = \begin{pmatrix} 11 \\ 00 \end{pmatrix} x\mathbf{u} \begin{pmatrix} 00 \\ 10 \end{pmatrix} + \begin{pmatrix} 20 \\ 00 \end{pmatrix} \mathbf{u} + \begin{pmatrix} 00 \\ 10 \end{pmatrix} \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \begin{pmatrix} 10 \\ 00 \end{pmatrix} \quad 0 \quad (9)$$

Кольцо матриц $R = P_{2,2}$ простое, то есть любой ненулевой двусторонний идеал кольца R совпадает с R . Следовательно, двусторонний идеал кольца R , порожденный произвольной ненулевой матрицей, совпадает с R . Поэтому с помощью отрезка, имеющего первый ненулевой элемент на k -м месте, можно обнулить k -й элемент произвольного отрезка. Так как отрезок $x^2u(0, 7)$ имеет длину 6, вычитая из него некоторую двустороннюю линейную комбинацию отрезков (1)–(9), можно получить нулевой отрезок. Отсюда следует, что отрезок $x^2u(0, 7)$ является двусторонней линейной комбинацией отрезков $xu(0, 7)$ и $u(0, 7)$, и, значит, двусторонний ранг отрезка $u(0, 7)$ не превосходит 2.

Попытки представить отрезок $xu(0, 7)$ в виде $xu(0, 7) = \sum B_i u(0, 7) C_i$, где B_i, C_i — матрицы размеров 2×2 , не привели к успеху. Поэтому, скорее всего, двусторонний ранг отрезка $u(0, 7)$ равен 2 (убедиться в этом можно, заполнив 211 таблиц). Пользуясь соотношениями (1)–(9), найдем двусторонний закон рекурсии для отрезка $u(0, 7)$:

$$x^2u(0, 7) = x^2\mathbf{u} \quad \begin{pmatrix} 10 \\ 00 \end{pmatrix} \begin{pmatrix} 22 \\ 11 \end{pmatrix} \begin{pmatrix} 10 \\ 22 \end{pmatrix} \begin{pmatrix} 10 \\ 20 \end{pmatrix} \begin{pmatrix} 21 \\ 00 \end{pmatrix} \begin{pmatrix} 10 \\ 01 \end{pmatrix} \quad (10)$$

$$x^2\mathbf{u} + \begin{pmatrix} 20 \\ 00 \end{pmatrix} \mathbf{u} \begin{pmatrix} 10 \\ 00 \end{pmatrix} \quad 0 \quad \begin{pmatrix} 22 \\ 11 \end{pmatrix} \begin{pmatrix} 00 \\ 22 \end{pmatrix} \begin{pmatrix} 20 \\ 20 \end{pmatrix} \begin{pmatrix} 11 \\ 00 \end{pmatrix} \begin{pmatrix} 00 \\ 01 \end{pmatrix} \quad (11)$$

$$x^2\mathbf{u} + \begin{pmatrix} 20 \\ 00 \end{pmatrix} \mathbf{u} \begin{pmatrix} 10 \\ 00 \end{pmatrix} + \begin{pmatrix} 21 \\ 12 \end{pmatrix} \mathbf{u} \quad 0 \quad 0 \quad \begin{pmatrix} 20 \\ 02 \end{pmatrix} \begin{pmatrix} 12 \\ 01 \end{pmatrix} \begin{pmatrix} 20 \\ 21 \end{pmatrix} \begin{pmatrix} 10 \\ 21 \end{pmatrix} \quad (12)$$

$$(12) + (6) + \begin{pmatrix} 01 \\ 00 \end{pmatrix} (6) \begin{pmatrix} 00 \\ 10 \end{pmatrix} \quad 0 \quad 0 \quad 0 \quad \begin{pmatrix} 02 \\ 01 \end{pmatrix} \begin{pmatrix} 00 \\ 01 \end{pmatrix} \begin{pmatrix} 11 \\ 11 \end{pmatrix} \quad (13)$$

$$(13) + \begin{pmatrix} 20 \\ 10 \end{pmatrix} (7) \begin{pmatrix} 01 \\ 00 \end{pmatrix} \quad 0 \quad 0 \quad 0 \quad 0 \quad \begin{pmatrix} 02 \\ 02 \end{pmatrix} \begin{pmatrix} 11 \\ 11 \end{pmatrix} \quad (14)$$

$$(14) + \begin{pmatrix} 10 \\ 10 \end{pmatrix} (8) \begin{pmatrix} 01 \\ 00 \end{pmatrix} \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \begin{pmatrix} 10 \\ 10 \end{pmatrix} \quad (15)$$

$$(15) + \begin{pmatrix} 20 \\ 20 \end{pmatrix} (9) \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad (16)$$

Соотношение (16) вместе с предыдущими дает следующий двусторонний закон рекурсии для отрезка $u = u(0, 7)$:

$$\begin{aligned} x^2 u + \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} u \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} u + \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x u + \begin{pmatrix} 2 & 0 \\ 2 & 0 \end{pmatrix} u \\ + \begin{pmatrix} 2 & 1 \\ 0 & 0 \end{pmatrix} u \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} x u \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} u \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \\ + \begin{pmatrix} 2 & 2 \\ 1 & 1 \end{pmatrix} x u \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 0 \\ 2 & 0 \end{pmatrix} u \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 1 & 2 \\ 2 & 1 \end{pmatrix} u \begin{pmatrix} 0 & 2 \\ 0 & 0 \end{pmatrix} \\ + \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix} x u \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 2 & 0 \\ 2 & 0 \end{pmatrix} u \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix} x u \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \\ + \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} u \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} = \bar{0}. \end{aligned}$$

Приводя подобные слагаемые (например, второе и шестое, третье и пятое и т.д.), получим

$$\begin{aligned} x^2 u + \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x u + \begin{pmatrix} 2 & 0 \\ 2 & 2 \end{pmatrix} x u \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + \begin{pmatrix} 2 & 0 \\ 1 & 2 \end{pmatrix} x u \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \\ + \begin{pmatrix} 1 & 1 \\ 0 & 2 \end{pmatrix} u + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} u \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} u \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + \begin{pmatrix} 2 & 1 \\ 2 & 2 \end{pmatrix} u \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = \bar{0}. \end{aligned}$$

Этот двусторонний закон рекурсии можно также записать в виде левого характеристического многочлена отрезка $u(0, 7)$ над левым модулем ${}_S R$, где $S = R \otimes R^{op}$:

$$\begin{aligned} \mu_{lr}(x) = x^2 - \left(\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \otimes E + \begin{pmatrix} 2 & 0 \\ 2 & 2 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + \begin{pmatrix} 2 & 0 \\ 1 & 2 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \right) x \\ - \left(\begin{pmatrix} 1 & 1 \\ 0 & 2 \end{pmatrix} \otimes E + \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 2 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} \otimes \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + \begin{pmatrix} 2 & 1 \\ 2 & 2 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} \right). \end{aligned}$$

Таким образом, левый ранг отрезка $u(0, 7)$ над кольцом R равен 3, правый ранг равен 4, а двусторонний ранг равен 2 или 1.

Данный пример показывает, что задача нахождения двустороннего ранга отрезка над кольцом R с помощью алгоритма нахождения левого ранга и левого минимального многочлена этого отрезка над модулем ${}_S R$, где $S = R \otimes R^{op}$, оказывается весьма трудоемкой. Это объясняется тем, что кольцо $R \otimes R^{op}$ может быть устроено

значительно сложнее, чем кольцо R . В связи с этим представляет интерес отыскание более быстрого или усовершенствование предложенного алгоритма нахождения двустороннего ранга отрезка над кольцом R , а в более общей ситуации — алгоритма нахождения ранга и двустороннего закона рекурсии отрезка над бимодулем ${}_A M_B$.

6. Краткий обзор литературы

Впервые алгоритм Берлекэмпа–Мессе был изложен Берлекэмпом в монографии [7] в качестве алгоритма декодирования кодов БЧХ над конечным полем. До этого для декодирования кодов БЧХ использовался алгоритм Петерсона–Горенштейна–Цирлера [83, 49, 4, 15]. Мессе [73] показал (см. также [116]), что алгоритм Берлекэмпа можно интерпретировать как алгоритм построения многочлена наименьшей степени, порождающего заданный отрезок последовательности над полем. Эта интерпретация оказалась очень полезной, и теперь алгоритм называют алгоритмом Берлекэмпа–Мессе.

Исходный алгоритм Берлекэмпа достаточно сложен. Рядом авторов впоследствии этот алгоритм был подробно исследован [8, 18, 32, 40, 51, 56, 57, 115]. Были также опубликованы различные, более наглядные, версии алгоритма. В работах [9, 16, 17, 21, 29, 34, 35, 37, 38, 77] алгоритм Берлекэмпа–Мессе излагается с использованием алгоритма Евклида деления многочленов над полем. Эквивалентность алгоритма Берлекэмпа–Мессе и алгоритма Евклида доказана в [34]. В работах [25, 26, 30, 61, 78, 113] алгоритм Берлекэмпа–Мессе излагается с помощью непрерывных дробей и рациональной аппроксимации, в [66, 116] с использованием свойств линейных рекуррентных последовательностей над полем. В работах [14, 74, 80, 86] задача нахождения степени минимального многочлена периодической последовательности над полем с приложениями декодирования различных классов кодов решается с использованием дискретного преобразования Фурье. В [48] приводится алгоритм нахождения минимального многочлена двоичной последовательности специального вида периода 2^n .

В работах [19, 52] рассматривается алгоритм решения теплицевых систем линейных уравнений с трудоемкостью $O(l \log^2 l)$, позволяющий снизить трудоемкость алгоритма Берлекэмпа–Мессе до такой же величины, однако, в последующем сообщении [50] указывалось, что обоснование предложенного алгоритма являлось неполным. Имеются вероятностные версии алгоритма Берлекэмпа–Мессе [27, 62, 63, 64, 114], а также версии, допускающие распараллеливание [9, 27, 28, 35, 62, 63, 98, 115]. В монографиях алгоритм Берлекэмпа–Мессе излагается, например, в [2, 3, 4, 7, 15, 16, 17, 55, 66, 76].

Рассматривались различные обобщения алгоритма Берлекэмпа–Мессе над полем. Так, в [24, 33, 36, 38, 37, 39, 65, 80] приводятся алгоритмы построения многочлена наименьшей степени, аннулирующего одновременно несколько отрезков над полем. В [70, 106] даются алгоритмы нахождения рангов (степеней минимальных многочленов) всех подотрезков заданного отрезка. В работах Сакаты [87, 88, 89, 90, 91, 92, 94, 95, 96, 97, 98], а также в [43, 46], рассматриваются алгоритмы построения аннулирующего идеала nD -последовательности, использующие теорию базисов идеалов в кольце многочленов от нескольких переменных (базисов Гребнера) и являющиеся обобщением алгоритма Берлекэмпа–Мессе на многомерные последовательности. В [99] рассматриваются последовательности над конусами и

цилиндрами. В [58] исследовалась задача нахождения наикратчайшего, возможно нелинейного, регистра, порождающего заданный отрезок.

Известны обобщения алгоритма Берлекэмп–Мессе для отрезков над различными кольцами. Алгоритм Берлекэмп–Мессе для колец матриц над полем рассматривался в [27, 31, 62, 63, 68, 112]. В работах [31, 112] рассматривалась задача нахождения аппроксимации Падэ для матричных последовательностей. Предлагаемые алгоритмы основаны на подборе линейно независимых столбцов многочленов над полем, удовлетворяющих специальным индуктивным условиям, которые объединяются в полиномиальную матрицу над полем, дающую решение задачи аппроксимации. Однако, не подсчитана трудоемкость алгоритмов. Кроме того, при использовании этих алгоритмов для нахождения многочлена, порождающего заданный отрезок над кольцом матриц, не всегда получается унитарный многочлен (см. пример в статье [31], в котором как старший, так и свободный коэффициент полученного многочлена являются вырожденными матрицами). Таким образом, данные алгоритмы находят не минимальный многочлен отрезка, а многочлен, аннулирующий этот отрезок. В работах [27, 62, 63] излагается допускающий распараллеливание метод решения разреженных систем линейных уравнений над конечным полем P (с приложениями к разложению больших чисел на множители), в процессе которого требуется находить минимальный многочлен матрицы. Соответствующий алгоритм, позволяющий находить минимальный многочлен матрицы B над конечным полем $P = GF(q)$, назван алгоритмом Берлекэмп–Мессе. Этот алгоритм вероятностный и приводит к результату с вероятностью, не меньшей, чем $1 - (2 \operatorname{rank} B + 1)/q$, а его обоснование остается открытой задачей (см. [63]). Кроме того, алгоритм не работает при некоторых специальных условиях на собственные значения матрицы B .

В работе [85] рассматривается алгоритм Берлекэмп–Мессе над кольцами вычетов $\mathbf{Z}_m$. Основу работы составляет алгоритм Евклида в кольце $\mathbf{Z}_{p^N}[x]$. Используя терминологию нашей работы, этот алгоритм можно охарактеризовать как одновременное заполнение N таблиц, причем вычисления в каждой таблице аналогичны алгоритму Евклида над полем $\mathbf{Z}_p$, и для вычислений многочленов в одной таблице используются многочлены из всех N таблиц. Именно эта идея работы [85] обобщена нами на случай произвольного конечного кольца R , при этом N — число классов ненулевых лево-ассоциированных элементов кольца R .

В [93] приводится обобщение алгоритма работы [85] на nD -последовательности. Используется тот факт, что кольцо $\mathbf{Z}_m[x]$ является кольцом с редукцией, и в нем имеется алгоритм построения базиса Гребнера.

В работах [46, 47, 79] приводится алгоритм Берлекэмп–Мессе над факториальной коммутативной областью целостности R . Используется возможность вложения R в поле частных, а также существование и единственность разложения на множители в R . Автором рассматривался алгоритм Берлекэмп–Мессе над коммутативными артиновыми кольцами главных идеалов, частным случаем которых являются кольца Галуа и кольца вычетов $\mathbf{Z}_m$. Задача нахождения минимального многочлена отрезка сводится к случаю локального кольца R с максимальным идеалом J , и затем используется метод подъема минимального многочлена над полем вычетов $\bar{R} = R/J$, который ищется с помощью алгоритма Берлекэмп–Мессе над полем, до минимального многочлена отрезка над кольцом R . Эти результаты будут опубликованы позднее. В работе [1] излагается алгоритм Берлекэмп–Мессе для случая произвольного конечного коммутативного кольца с единицей. Алгоритм

данной работы является его непосредственным обобщением.

Приложения алгоритма Берлекэмп-Мессе связаны с декодированием различных классов кодов (см. перечисленные выше монографии и работы [14, 79, 80, 104]), включая коды Рида-Соломона [13, 28, 65, 115], коды БЧХ [12, 13, 43, 21, 23, 24, 40, 42, 104, 107, 115, 116], циклические [39, 40, 107] и обобщенные циклические [110] коды, альтернантные коды [13, 43, 54, 71], коды Гоппы [71, 72, 77, 82, 103, 108, 109] и алгебро-геометрические коды [67, 69, 81, 84, 86, 99, 100, 101, 102], nD -циклические коды [22, 86, 87, 93, 94, 98], коды над кольцами [93]. Алгоритм Берлекэмп-Мессе используется для решения ганкелевых и теплицевых [9, 17, 19, 50, 52, 59, 66, 68, 97], разреженных [27, 62, 63, 64, 114] и общих [79] систем линейных уравнений, при поиске рациональной аппроксимации функций [12, 44, 52] и, в частности, аппроксимации Падэ [9, 19, 20, 31, 41, 50, 52, 60, 77, 79, 112]. Задачу аппроксимации Падэ можно также сформулировать на языке теории систем как задачу нахождения линейной системы с наименьшим числом состояний (задача минимальной реализации) по заданным марковским параметрам и моментам, см., например, [10, 11, 31, 60, 79, 112]. Алгоритм Берлекэмп-Мессе используется также в криптографии и при тестировании псевдослучайных последовательностей [70, 75, 105, 106], и для быстрого вычисления в конечных полях [111].

Список литературы

1. Куракин В. Л. Алгоритм Берлекэмп-Мессе над конечными кольцами. *Труды 6-й Международной конф. по алгебраической и комбинаторной теории кодирования*. Псковский госунив., Псков, 1998, 161–165.
2. Лидл Р., Нидеррайтер Г. *Конечные поля*. Т. 1, 2. Мир, Москва, 1988.
3. Мак-Вильямс Ф. Д., Слоэн Н. Д. А. *Теория кодов, исправляющих ошибки*. Связь, Москва, 1979.
4. Питерсон У., Уэлдон Э. *Коды, исправляющие ошибки*. Мир, Москва, 1976.
5. Фейс К. *Алгебра: кольца, модули, категории*. Т. 2. Мир, Москва, 1979.
6. Berlekamp E. R. Nonbinary BCH decoding. *International Symposium on Information Theory*. San Remo, Italy, 1962.
7. Berlekamp E. R. *Algebraic Coding Theory*. McGraw-Hill, New York, 1968.
8. Berlekamp E. R., Fredricksen E. M., Proto R. C. Minimum conditions for uniquely determining the generator of a linear sequence. *Utilitas Math.* (1974) **5**, 305–315.
9. Bini D. Pan V. Improved parallel computations with Toeplitz-like and Hankel-like matrices. *Linear Algebra Appl.* (1993) **188/189**, 3–29.
10. Bistritz Y. Nested bases of invariants for minimal realization of finite matrix sequences. *SIAM J. Control Opt.* (1983) **21**, 804–821.
11. Bistritz Y., Shaked U. Discrete multivariable system approximations by minimal Padé type state models. *IEEE Trans. Circuits and Systems* (1984) **31**, 382–390.
12. Blackburn S. R. The Welch-Berlekamp and Berlekamp-Massey algorithms. *IEEE Int. Symp. Inform. Theory*, Whistler, BC, Canada, 1995.
13. Blackburn S. R. Chambers W. G. Some remarks on an algorithm of Fitzpatrick. *IEEE Trans. Inform. Theory* (1996) **42**, 1269–1271.
14. Blahut R. E. Transform techniques for error control codes. *IBM J. Res. Develop.* (1979) **23**, 299–315.

15. Blahut R. E. *Theory and Practice of Error Control Codes*. Addison-Wesley, Reading, Mass., 1983.
16. Blahut R. E. *Fast Algorithms for Digital Signal Processing*. Addison-Wesley, Reading, Mass., 1985.
17. Blahut R. E. *Algebraic Methods for Signal Processing and Communications Coding*. Springer, Berlin, 1992.
18. Bourne C., Piper F. Finite subsequences of shift register sequences. *J. Appl. Math.* (1985) **35**, 13–22.
19. Brent R. P., Gustavson F. G., Yun D. Y. Fast solution of Toeplitz systems of equations and computation of Padé approximants. *J. Algorithms* (1980) **1**, 259–295.
20. Bultheel A. Recursive algorithms for non-normal Padé tables. *SIAM J. Appl. Math.* (1980) **39**, 106–118.
21. Camion P. An iterative Euclidean algorithm. *Lect. Notes Comput. Sci.* (1987) **365**, 88–128.
22. Chabanne H., Norton G. H. The n -dimensional key equation and a decoding application. *IEEE Trans. Inform. Theory* (1994) **40**, 200–203.
23. Chen C. L. High-speed decoding of BCH codes. *IEEE Trans. Inform. Theory* (1981) **27**, 254–266.
24. Chen P. H. Multisequence linear shift-register synthesis and its application to BCH decoding. *IEEE Trans. Commun.* (1976) **24**, 438–440.
25. Cheng U. On the continued fractions and Berlekamp's algorithm. *IEEE Trans. Inform. Theory* (1984) **30**, 541–544.
26. Conan J. A recursive procedure for the solution of the minimal partial realization problem for scalar rational sequences. *Rev. Roumaine Math. Pures Appl.* (1985) **30**, 625–645.
27. Coopersmith D. Solving homogeneous linear equations over $GF(2)$ via block Wiedemann algorithm. *Math. Comput.* (1994) **62**, 333–350.
28. Dabiri D., Blake I. F. Fast parallel algorithms for decoding Reed–Solomon codes based on remainder polynomials. *IEEE Trans. Inform. Theory* (1995) **41**, 873–885.
29. Dai Z. D., Wan Z. X. A relationship between the Berlekamp–Massey and the Euclidean algorithms for linear feedback shift-register synthesis. *Acta. Math. Sinica. New Ser.* (1988) **4**, 55–63.
30. Dai Z. D., Zeng K. C. Continued fractions and the Berlekamp–Massey algorithm. *Lect. Notes Comput. Sci.* (1990) **453**, 23–31.
31. Dickinson B. W., Morf M., Kailath T. A minimal realization algorithm for matrix sequences. *IEEE Trans. Automat. Control* (1974) **19**, 31–38.
32. Dillon J. F., Morris R. A. On a paper of Berlekamp, Fredricksen and Proto. *Utilitas Math.* (1974) **5**, 317–321.
33. Ding C. Proof of Massey's conjectured algorithm. *Lect. Notes Comput. Sci.* (1988) **330**, 345–349.
34. Dornstetter J. L. On the equivalence between Berlekamp' and Euclid's algorithms. *IEEE Trans. Inform. Theory* (1987) **33**, 428–431.
35. Eastman W. L. Inside Euclid's algorithm. In: *Coding Theory and Design Theory, Part I: Coding Theory*. Springer, Berlin, 1990, 113–127.
36. Feng G. L., Tzeng K. K. An iterative algorithm of shift-register synthesis for multiple sequence. *Sci. Sinica. Ser. A* (1985) **28**, 1222–1232.

37. Feng G. L., Tzeng K. K. Euclidean algorithm for multisequence shift-register synthesis. *Sci. Sinica. Ser. A* (1987) **30**, 535–547.
38. Feng G. L., Tzeng K. K. G. A generalized Euclidean algorithm for multisequence shift-register synthesis. *IEEE Trans. Inform. Theory* (1989) **35**, 584–594.
39. Feng G. L., Tzeng K. K. A generalization of the Berlekamp–Massey algorithm for multisequence shift-register synthesis with application to decoding cyclic codes. *IEEE Trans. Inform. Theory* (1991) **37**, 1274–1287.
40. Feng G. L., Tzeng K. K. Decoding cyclic and BCH codes up to actual minimum distance using nonrecurrent syndrome dependence relations. *IEEE Trans. Inform. Theory* (1991) **37**, 1716–1723.
41. Fitzpatrick P. A theoretical basis for Pade approximation. *Irish Math. Soc. Bull.* (1993) **30**, 6–17.
42. Fitzpatrick P. New time-domain errors-and-erasures decoding algorithm for BCH codes. *Electron. Lett.* (1994) **30**, 110–111.
43. Fitzpatrick P. On the key equation. *IEEE Trans. Inform. Theory* (1995) **41**, 1290–1302.
44. Fitzpatrick P. On the scalar rational interpolation problem. *Mathematics of Control, Signals, and Systems* (1996) **9**, 352–369.
45. Fitzpatrick P., Norton G. H. Linear recurrence relations and an extended subresultant algorithm. *Lect. Notes Comput. Sci.* (1989) **388**, 232–243.
46. Fitzpatrick P., Norton G. H. Finding a basis for the characteristic ideal of an n -dimensional linear recurring sequence. *IEEE Trans. Inform. Theory* (1990) **36**, 1480–1487.
47. Fitzpatrick P., Norton G. H. The Berlekamp–Massey algorithm and linear recurring sequences over a factorial domain. *Applicable Algebra in Engineering, Communication and Computing* (1995) **6** (1995), 309–323.
48. Games R. A., Chan A. H. A fast algorithm for determining the complexity of a binary sequence with period 2^n . *IEEE Trans. Inform. Theory* (1983) **29**, 144–146.
49. Gorenstein D. C., Zierler N. A class of error correcting codes in p^m symbols. *J. Soc. Indust. Appl. Math.* (1961) **9**, 207–214.
50. Gragg W. B., Gustavson F. G., Warner D. D., Yun D. Y. Y. On fast computation of superdiagonal Padé fractions. *Math. Programming Stud.* (1982) **18**, 39–42.
51. Gustavson F. G. Analysis of the Berlekamp–Massey linear feedback shift-register synthesis algorithm. *IBM J. Res. Develop.* (1976) **20**, 204–212.
52. Gustavson F. G., Yun D. Y. Y. Fast algorithms for rational Hermite approximation and solution of Toeplitz systems. *IEEE Trans. Circuits and Systems* (1979) **26**, 750–755.
53. Hammons A. R., Kumar P. V., Calderbrank A. R., Sloane N. J. A., Sole P. The $\mathbb{Z}_4$ -linearity of Kerdock, Preparata, Goethals and related codes. *Bull. Amer. Math. Soc.* (1993) **29**, 218–222.
54. Helgert N. J. Decoding of alternant codes. *IEEE Trans. Inform. Theory* (1977) **23**, 513–514.
55. Hoffman D. G., Leonard D. A., Lindner C. C., Phelps K. T., Rodger C. A., Wall J. R. *Coding Theory*. Marcel Dekker, New York, 1991.
56. Imamura K. Two recursive algorithms for solving the scalar partial realization problem. *IEEE ISCAS 85*, 823–824.
57. Imamura K., Yoshida W. A simple derivation of the Berlekamp–Massey algorithm and some applications. *IEEE Trans. Inform. Theory* (1987) **33**, 146–150.

58. Jansen C. J. A., Boeke D. E. The shortest feedback shift-register that can generate a given sequence. *Lect. Notes Comput. Sci.* (1990) **435**, 90–99.
59. Jonckheere E., Ma C. A simple Hankel interpretation of the Berlekamp–Massey algorithm. *Linear Algebra Appl.* (1989) **125**, 65–76.
60. Jonckheere E., Ma C. Combined sequence of Markov parameters and moments in linear systems. *IEEE Trans. Automat. Control* (1989) **34**, 379–382.
61. Kalman R. T. On partial realization, transfer functions, and canonical forms. *Acta Poly. Scand. Math. Comput.* (1979) **31**, 9–32.
62. Kaltofen E. Analysis of Coopersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems. *Lect. Notes Comput. Sci.* (1993) **673**, 195–212.
63. Kaltofen E. Analysis of Coopersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems. *Math. Comput.* (1995) **64**, 777–806.
64. Kaltofen E., Saunders B. D. On Wiedemann's method of solving sparse linear systems. *Lect. Notes Comput. Sci.* (1991) **539**, 29–38.
65. Kamiya N. On multisequence shift register synthesis and generalized minimum distance decoding of Reed–Solomon codes. *Finite Fields Appl.* (1995) **1**, 440–457.
66. Kurakin V. L., Kuzmin A. S., Mikhalev A. V., Nechaev A. A. Linear recurrences over rings and modules. *J. Math. Sci.* (1995) **76**, 2793–2915.
67. Kurihara M., Mizuno H. Fast algorithm for finding an error locator function for algebraic-geometric codes from elliptic and hyperelliptic curves: $O(n^2)$. *Bull. Univ. Electro-Comm.* (1992) **5**, 1–11.
68. Labahn G. Inversion components of block Hankel-like matrices. *Linear Algebra Appl.* (1992) **177**, 7–48.
69. Leonard D. A. Error-locator ideals for algebraic-geometric codes. *IEEE Trans. Inform. Theory* (1995) **41**, 819–824.
70. Lin X. D., Cai C. N. Sequences with controllable linear complexity profile. *Chinese Sci. Bull.* (1990) **35**, 867–869.
71. Loeloeian M., Conan J. A transform approach to Goppa codes. *IEEE Trans. Inform. Theory* (1987) **33**, 105–115.
72. Mandelbaum D. M. A method for decoding of generalized Goppa codes. *IEEE Trans. Inform. Theory* (1977) **23**, 137–140.
73. Massey J. L. Shift-register synthesis and BCH decoding. *IEEE Trans. Inform. Theory* (1969) **15**, 122–127.
74. Massey J. L., Schaub T. Linear complexity in coding theory. *Lect. Notes Comput. Sci.* (1988) **311**, 19–32.
75. McCurley K. S. Odds and ends from cryptology and computational number theory. *Cryptology and Computational Number Theory* (1989), 145–166.
76. McEliece R. J. *The Theory of Information and Coding*. Addison-Wesley, Reading, Mass., 1977.
77. McEliece R. J., Shearer J. B. A property of Euclid's algorithm and application to Padé approximation. *SIAM J. Appl. Math.* (1978) **34**, 611–615.
78. Mills W. H. Continued fractions and linear recurrences. *Math. Comput.* (1975) **29**, 173–180.
79. Norton G. On the minimal realization of a finite sequence. *J. Symbolic Comput.* (1995) **20**, 93–115.

80. Norton G. Some decoding applications of minimal realization. *Lect. Notes Comput. Sci.* (1995) **1025**, 53–62.
81. O'Sullivan M. E. Decoding of codes defined by a single point on a curve. *IEEE Trans. Inform. Theory* (1995) **41**, 1709–1719.
82. Patterson N. J. The algebraic decoding of Goppa codes. *IEEE Trans. Inform. Theory* (1975) **21**, 203–207.
83. Peterson W. W. Encoding and error-correction procedures for the Bose–Chaudhuri codes. *IEEE Trans. Inform. Theory* (1960) **6**, 459–470.
84. Porter S. C., Shen B. Z., Pellikaan R. Decoding geometric Goppa codes using an extra place. *IEEE Trans. Inform. Theory* (1992) **38**, 1663–1676.
85. Reeds J. A., Sloane N. J. A. Shift-register synthesis (modulo m). *SIAM J. Comput.* (1985) **14**, 505–513.
86. Saints K., Heegard C. Algebraic-geometric codes and multidimensional cyclic codes: a unified theory and algorithms for decoding using Gröbner bases. *IEEE Trans. Inform. Theory* (1995) **41**, 1733–1751.
87. Sakata S. On determining the independent set for doubly periodic arrays and encoding two-dimensional cyclic codes and their duals. *IEEE Trans. Inform. Theory* (1981) **27**, 556–565.
88. Sakata S. Higher-dimensional extension of the Berlekamp–Massey algorithm and Gröbner bases. In: *Proc. Symposium held at the Research Institute for Math. Sciences*. Kyoto Univ., Kyoto, 1986, pp. 52–66.
89. Sakata S. Finding a minimal set of linear recurring relations capable of generating a given finite two-dimensional array. *J. Symbolic Comput.* (1988) **5**, 321–337.
90. Sakata S. Synthesis of two-dimensional linear feedback shift-registers and Groebner bases. *Lect. Notes Comput. Sci.* (1989) **356**, 394–407.
91. Sakata S. n -dimensional Berlekamp–Massey algorithm for multiple arrays and construction of multivariate polynomials with preassigned zeros. *Lect. Notes Comput. Sci.* (1989) **357**, 356–376.
92. Sakata S. Extension of the Berlekamp–Massey algorithm to N dimensions. *Inform. and Comput.* (1990) **84**, 207–239.
93. Sakata S. Two-dimensional shift register synthesis and Groebner bases for polynomial ideals over an integer residue ring. *Discr. Appl. Math.* (1991) **33**, 191–203.
94. Sakata S. Decoding binary $2D$ cyclic codes by the $2D$ Berlekamp–Massey algorithm. *IEEE Trans. Inform. Theory* (1991) **37**, 1200–1203.
95. Sakata S. A Gröbner basis and a minimal polynomial set of a finite nD array. *Lect. Notes Comput. Sci.* (1991) **508**, 280–291.
96. Sakata S. Finding a minimal polynomial vector set of a vector of nD arrays. *Lect. Notes Comput. Sci.* (1991) **539**, 414–425.
97. Sakata S. On the minimal partial realization of $2D$ discrete linear shift-invariant systems. *IEEE Trans. Automat. Control* (1991) **36**, 984–988.
98. Sakata S. Decoding $2D$ cyclic codes by paralleling the $2D$ Berlekamp–Massey algorithm. *CISM Courses and Lectures* (1993) **339**, 277–289.
99. Sakata S. Shift register synthesis on convex cones and cylinders and fast decoding of general one-point AG codes. *Bull. Univ. Electro-Comm.* (1995) **8**, 187–203.
100. Sakata S. Fast erasure-and-error decoding of any one-point AG codes up to the Feng–Rao bound. *Bull. Univ. Electro-Comm.* (1996) **9**, 39–57.

101. Sakata S., Jensen H. E., Høholdt T. Generalized Berlekamp–Massey decoding of algebraic-geometric codes up to half the Feng–Rao bound. *IEEE Trans. Inform. Theory* (1995) **41**, 1762–1768.
102. Sakata S., Justesen J., Madelung Y., Jensen H. E., Høholdt T. A fast decoding method of AG codes from Miura–Kamiya curves C_{ab} up to half the Feng–Rao bound. *Finite Fields Appl.* (1995) **1**, 83–101.
103. Sarwate D. V. On the complexity of decoding Goppa codes. *IEEE Trans. Inform. Theory* (1977) **23**, 515–516.
104. Sendrier N. The polynomial of correctable patterns. *CISM Courses and Lectures* (1993) **339**, 291–303.
105. Sloane N. J. A. Encrypting by random rotations. *Lect. Notes Comput. Sci.* (1983) **149**, 71–128.
106. Stephens N. M. The zero-square algorithm for computing linear complexity profiles. In: *Inst. Math. Appl. Conf. Ser.* Vol. 33, Oxford Univ. Press, New York, 1992, 259–271.
107. Stevens P. Extension of the BCH decoding algorithm to decode binary cyclic codes up to their maximum error-correction capacities. *IEEE Trans. Inform. Theory* (1988) **34**, 1332–1340.
108. Sugiyama Y., Kasahara M., Hirasawa S., Namekawa T. A method for solving the key equation for decoding Goppa codes. *Inform. Control* (1975) **27**, 87–99.
109. Sugiyama Y., Kasahara M., Hirasawa S., Namekawa T. Further results on Goppa codes and their applications to constructing efficient binary codes. *IEEE Trans. Inform. Theory* (1976) **22**, 518–526.
110. Tanner R. M. A transform theory for a class of group-invariant codes. *IEEE Trans. Inform. Theory* (1988) **34**, 752–775.
111. Thiong Ly J. A. Note for computing the minimum polynomial of elements in large finite fields. *Lect. Notes Comput. Sci.* (1989) **388**, 185–192.
112. Van-Barel M., Bultheel A. Minimal Pade-sense matrix approximations around $s = 0$ and $s = \infty$. In: *Mathematics and its Applications*. Vol. 43, Reidel, Dordrecht, 1988, 143–154.
113. Welch L. R., Scholtz R. A. Continued fractions and Berlekamp’s algorithm. *IEEE Trans. Inform. Theory* (1979) **25**, 19–27.
114. Wiedemann D. H. Solving sparse linear equations over finite fields. *IEEE Trans. Inform. Theory* (1986) **32**, 54–62.
115. Zarowski C. J. A divisionless form of the Schur Berlekamp–Massey algorithm. *Lect. Notes Comput. Sci.* (1994) **793**, 38–44.
116. Zierler N. Linear recurring sequences and error-correcting codes. In: *Error Correcting Codes*. (H. B. Mann, ed.), Wiley, New York, 1968, 47–59.

Статья поступила 14.07.1998.