УДК 519.725

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ ОДНОГО МЕТОДА ПОРОГОВОГО РАЗДЕЛЕНИЯ ДАННЫХ

С. Е. Гурова, Н. С. Могилевская
Институт механики, математики и компьютерных наук имени И. И. Воровича Южного федерального университета (г. Ростов-на-Дону, Российская Федерация)

Аннотация. Одним из способов организации отказоустойчивых хранилищ данных является использование (n,k)-пороговых методов разделения данных, где k<n. Исходные данные D разбиваются на n частей таким образом, что по любым k частям можно восстановить D. Рассмотрен пороговый метод ТХП разделения данных, основанный на работе с матрицами типа матрицы Вандермонда и полями Галуа простой мощности, предложены различные режимы его работы. Получены оценки размера вспомогательных структур данных, возникающих при организации их хранения с использованием метода. Цель данной работы состоит в переносе (*n*,*k*)-порогового метода ТХП разделения данных на случай полей Галуа характеристики 2. Построено программное средство, реализующее метод ТХП различных полей Галуа. Проведены эксперименты, показавшие, что применение метода ТХП над полями Галуа характеристики 2 сокращает объем хранимых данных.

Ключевые слова: система распределенного хранения, пороговый метод разделения, отказоустойчивое хранение данных, матрица Вандермонда, простые поля Галуа, поля Галуа характеристики 2.

IMPROVING THE EFFICIENCY OF ONE METHOD OF THRESHOLD DATA SEPARATION

Nadezhda S. Mogilevskaya, Svetlana E. Gurova
Institute of Mechanics, Mathematics and Computer Sciences named after I. I. Vorovich of the Southern Federal University (Rostov-on-Don, Russian Federation)

Abstract. One of the ways to organize fault-tolerant data warehouses is to use (n,k)-threshold methods of data separation, where k<n. The original data D is divided into n parts, so that for any k part it is possible to restore D. The paper considers the (n,k)-threshold method of data separation based on working with matrices of the Vandermond matrix type and Galois fields of simple power. Estimates of the size of auxiliary data structures that arise when organizing data storage using Galois fields of simple power are obtained. The method of data separation has been transferred to the case of Galois fields of characteristic 2. A software tool has been built that implements the method of data separation for arbitrary Galois fields. Experiments have been conducted that have shown that the use of the TP method over Galois fields of power 2 reduces the amount of stored data.

Keywords: distributed storage system, threshold separation method, fault-tolerant data storage, Vandermond matrix, simple Galois fields, Galois fields characteristics 2.

Введение. Один из популярных способов надежного хранения больших объемов данных состоит в использовании систем распределенного хранения [1–4]. Такие системы разделяют данные между множеством хранилищ, при этом данные возможно получить даже при недоступности какого-либо из накопителей. Для распределения данных на множество хранилищ могут быть использованы (*n*,*k*)-пороговые методы разделения данных. Такие методы разбивают входные данные D на *n* частей (долей) таким образом, что размер каждой части меньше размера входных данных, однако при этом данные D можно восстановить по произвольным *k* имеющимся частям.

Использование распределенного хранения может быть полезно не только для организации надежного хранения, но и для защиты данных от нарушения их конфиденциальности, так как злоумышленнику требуется перехватить k частей исходных данных D, которые могут храниться или передаваться независимо друг от друга. Примеры (n,k)-пороговых методов разделения данных описаны в работах [1–4].

В работе [1] рассмотрен (n,k)-пороговый метод разделения данных, основанный на работе с матрицами типа матрицы Вандермонда и полями Галуа простой мощности. Далее для удобства будем называть этот метод по первым буквам фамилий его авторов (Тормасов А. Г., Хасин М. А., Пахомов Ю. И.), а именно метод ТХП. Традиционно цифровые данные представляются в двоичном виде. При переводе данных из двоичного представления в представление элементами F_p, где p(>2) — простое число, возникает ряд проблем. Эти проблемы рассмотрены ниже.

Структура работы: в разделе 1 по материалам [1] описан ТХП метод порогового разделения данных, работающий с элементами простых полей Галуа, в разделе 2 указаны проблемы метода, возникающие из-за использования полей Галуа простой мощности, а также способы их решения. Раздел 3 посвящен описанию применения метода ТХП в случае полей Галуа характеристики 2. Раздел 4 содержит описание программной реализации метода ТХП разделения данных в полях произвольной мощности. В разделе 5 приведены некоторые результаты экспериментов и их обсуждение.

Основная часть. 1. Метод ТХП порогового разделения данных. Рассмотрим метод ТХП (n,k)-порогового разделения данных [1]. Данные, подлежащие разделению, рассматриваются как поток элементов поля Галуа F_p простой мощности p, при этом p>2. Для работы метода используется матрица

$$V = \begin{pmatrix} 1 & x_1 & \dots & x_1^{k-1} \\ 1 & x_2 & \dots & x_2^{k-1} \\ \dots & \dots & \dots & \dots \\ 1 & x_n & \dots & x_n^{k-1} \end{pmatrix}, \quad (1)$$

структурно похожая на матрицу Вандермонда, где x_i — порождающий элемент простого поля Галуа F_p, n и k — параметры метода ТХП, при этом $k<n$ [5]. Легко увидеть, что строки матрицы V являются линейно независимыми.

Рассмотрим основные шаги алгоритмов разделения и восстановления данных, при этом в алгоритмах не будем упоминать дополнительные действия, связанные с кодированием входных двоичных данных элементами полей Галуа F_p простой нечетной мощности.

Укрупненный алгоритм разделения данных.

Вход: параметры метода p, k, n, данные D в виде последовательности элементов F_p.

Выход: части (i, R_i), где $i=\overline{1,n}$ указывает на номер части, а вектор R_i содержит -тую часть разделенных данных.

Шаг 1. По формуле (1) построить матрицу V размером n строк k столбцов.

Шаг 2. Входные данные D записать поэлементно и построчно в матрицу M размером k строк и n столбцов.

Шаг 3. Вычислить

$$R = VM_1 = \begin{pmatrix} R_1 \\ \dots \\ R_n \end{pmatrix}.$$

Шаг 4. Для каждого $i = \overline{1,n}$ сформировать доли

$$D_i=R_i. \quad (2)$$

Конец.

Укрупненный алгоритм восстановления данных.

Вход: k различных долей (i_j, R_{i_j}), где $i_j \in [1, n]$, параметры метода k, n, p.

Выход: восстановленные данные D.

Шаг 1. Из k доступных долей $D_{i_j} = R_{i_j}$ исходных данных частей разделенных данных сформировать матрицу

$$R' = \begin{pmatrix} R_{i_1} \\ \dots \\ R_{i_k} \end{pmatrix}.$$

Шаг 2. Построить матрицу V

$$V = \begin{pmatrix} V_1 \\ \dots \\ V_n \end{pmatrix}.$$

Из строк матрицы V сформировать новую матрицу

$$V' = \begin{pmatrix} V_{i_1} \\ \dots \\ V_{i_k} \end{pmatrix}.$$

Шаг 3. Вычислить $M' = (V')^{-1}R'$.

Шаг 4. Восстановить исходные данные D, считывая элементы матрицы M' построчно.

Конец.

Рассмотрим пример работы метода с параметрами p=11, n=5, k=3. Зададим данные для разделения в виде двоичной последовательности:

D=10110001011010001011110101011000010111001000 1.

Элементами поля F_{11} являются числа от 0 до 10, следовательно, каждые три бита последовательности D могут быть переведены в элемент поля F_{11}. Таким образом, получаем представление входных данных в виде элементов поля F_{11}:

D=5,4,2,6,4,2,7,5,2,6,0,5,6,2,1.

В результате шагов 1–3 алгоритма разделения данных получаем (все вычисления проводятся по правилам поля F_{11}):

$$V = \begin{pmatrix} 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 6 & 3 \\ 1 & 7 & 5 \\ 1 & 10 & 1 \end{pmatrix}, M = \begin{pmatrix} 5 & 4 & 2 & 6 & 4 \\ 2 & 7 & 5 & 2 & 6 \\ 0 & 5 & 6 & 2 & 1 \end{pmatrix},$$

$$R = VM = \begin{pmatrix} 9 & 5 & 3 & 7 & 9 \\ 0 & 4 & 5 & 8 & 9 \\ 6 & 6 & 6 & 2 & 10 \\ 8 & 1 & 1 & 8 & 7 \\ 3 & 2 & 3 & 6 & 10 \end{pmatrix}.$$

Согласно шагу 4 алгоритма сформируем доли D_i, i=1, …, 5:

D_1=(9,5,3,7,9), D_2 = (0,4,5,8,9), D_3 = (6,6,6,2,10),

D_4 = (9,5,3,7,9), D_5 = (9,5,3,7,9).

Теперь предположим, что для восстановления данных доступны доли D_1, D_3 и D_4. Применим к ним алгоритм восстановления данных. Сначала построим матрицы R' и V':

$$R' = \begin{pmatrix} 9 & 5 & 3 & 7 & 9 \\ 6 & 6 & 6 & 2 & 10 \\ 8 & 1 & 1 & 8 & 7 \end{pmatrix} \qquad V' = \begin{pmatrix} 1 & 2 & 4 \\ 1 & 6 & 3 \\ 1 & 7 & 5 \end{pmatrix},$$

затем вычислим M':

$$M' = (V')^{-1}R' = \begin{pmatrix} 1 & 2 & 9 \\ 1 & 5 & 5 \\ 5 & 8 & 9 \end{pmatrix}\begin{pmatrix} 9 & 5 & 3 & 7 & 9 \\ 6 & 6 & 6 & 2 & 10 \\ 8 & 1 & 1 & 8 & 7 \end{pmatrix}=\begin{pmatrix} 5 & 4 & 2 & 6 & 4 \\ 2 & 7 & 5 & 2 & 6 \\ 0 & 5 & 6 & 2 & 1 \end{pmatrix}.$$

Легко увидеть, что $M = M'$, следовательно, исходные данные восстановлены корректно.

Сделаем ряд замечаний о методе ТХП.

Замечание 1. Сложность алгоритма разделения данных невысокая. Для построения матрицы V необходимо найти порождающие элементы поля F_p, однако это действие выполняется лишь однажды для заданных параметров метода. Таким образом, сложность алгоритма определяется трудоемкостью вычисления произведения матриц V и M.

Замечание 2. Наиболее трудоемким действием в алгоритме восстановления данных является вычисление обратной матрицы $(V')^{-1}$, следующим по сложности является вычисление произведения матриц $(V')^{-1}R'$.

2. Проблемы метода ТХП, связанные с использованием полей Галуа простой мощности, и способы их решения. Авторы ТХП предложили использовать в методе простые поля Галуа F_p нечетной мощности, т. е. где p — простое и $p > 2$. Очевидное преимущество такого решения состоит в том, что арифметика простых полей проста как для понимания, так и для выполнения. Фактически вычисления в простых полях сводятся к модульным операциям. Однако существуют и проблемы такого решения. Дело в том, что входные данные D обычно представляют собой последовательность бит, а для формирования матрицы M последовательность D необходимо последовательно разделять на отрезки по s бит таким образом, чтобы $2^s \approx p$, далее каждый из отрезков переводить в элемент поля F_p, формировать матрицу M, а затем выполнять алгоритмы разделения и восстановления данных.

Обозначим последовательность входных данных D, записанную в виде последовательности элементов поля F_p как $D_p = d_1, d_2, \ldots$. При этом могут возникнуть два случая: $2^s > p$ и $2^s < p$. Заметим, что авторы ТХП метода предлагают использовать случай $2^s > p$. Рассмотрим оба возможных варианта.

В случае $2^s > p$ в матрицу M могут быть записаны числа d_i, превосходящие мощность поля F_p. Для этого случая авторы метода ТХП предлагают в матрицу M записывать значения $d_i \bmod p$, а кроме этого, хранить вспомогательную структуру данных L_M, в которую записывать позиции всех элементов матрицы M, для которых исходное значение d_i>p. Следовательно, после работы алгоритма восстановления данных к элементам восстановленной матрицы M, номера которых записаны во вспомогательной структуре L_M, необходимо прибавить число p. Очевидно, что для восстановления данных необходимо знание L_M. Следовательно, структуру L_M следует хранить либо рядом с каждой долей, либо рядом с произвольными $n - k + 1$ долями (в этом случая среди любых k долей найдется хотя бы одна доля, снабженная структурой L_M).

Оценим размер структуры L_M. Так как по условию $2^s > p$, то существует $2^s - p$ комбинаций бит, которые не могут быть переведены в элементы поля F_p. Предположим, что при чтении исходных данных D все возможные комбинации из 2^s бит встречаются равновероятно, тогда частота, с которой встречаются «избыточные» элементы, $(2^s - p)/2^s$. Матрица M содержит nk элементов поля F_p, следовательно, в матрице M содержится $nk(2^s - p)/2^s$ таких элементов. В структуру L_M планируется записывать номера позиций матрицы M, которым соответствуют элементы, превосходящие p. Таким образом, для записи произвольного номера элемента матрицы M необходимо $\lceil \log_2 nk \rceil$ бит, где $\lceil . \rceil$ — округление вверх. Отсюда получаем среднюю оценку размера S_M структуры L_M, соответствующей одной матрице M:

$$S_M = \lceil \log_2 nk \rceil nk(2^s - p)/2^s,$$

где k, n — параметры метода,

p — мощность используемого поля Галуа.

В случае $2^s < p$ в матрицу M записываются числа d_i, значение которых меньше мощности поля F_p. Однако значения элементов матрицы R могут превосходить 2^s. В этой ситуации возможны два решения.

Первое из них состоит в том, что для хранения каждого элемента матрицы R нужно использовать число бит большее, чем s. В этом случае все возможные дополнительные структуры хранить не нужно, однако это решение отрицательно влияет на размер долей D_i=R_i (см. (2)).

Размер долей в этом случае будет

$$S_{R_i}^b = n\lceil \log_2 p \rceil,$$

а число бит, которые потребуются для хранения каждого элемента, как минимум s+1.

Второе решение состоит в том, чтобы записывать элементы R в доли с помощью s бит, но в доли D_i следует записывать не элементы матрицы $R = \{r_{ij}\}_{i,j=1,n}$, а результат операции $r_{ij}\ mod\ 2^s$. Для корректного восстановления данных для каждой доли D_i необходимо сформировать вспомогательную структуру L_{R_i}. В L_{R_i} следует помещать позиции R_i, в которых значения исходных элементов превышают 2^s. Очевидно, что L_{R_i} необходимо хранить для каждой доли D_i.

Оценим средний размер L_{R_i}. Так как по условию $2^s < p$, то существует $p - 2^s$ элементов поля F_p, которые не могут быть получены при чтении исходных данных D, однако могут встречаться в матрице R и, следовательно, в долях. Предположим, что все элементы поля F_p встречаются в матрице R равновероятно. Тогда возможные $(p - 2^s)$ значения элементов поля F_p, превосходящие p, встречаются в матрице R с частотой $(p - 2^s)/p$.

В структуру L_{R_i} планируется записывать номера позиций вектора R_i, которым соответствуют элементы, превосходящие 2^s. Вектор R_i состоит из n элементов, следовательно, в одном таком векторе находится примерно $n(p - 2^s)/p$ «особых» элементов. Для записи произвольного номера элемента матрицы R_i необходимо $\lceil \log_2 n \rceil$ бит. Отсюда получаем среднюю оценку размера S_{R_i} структуры L_{R_i}, соответствующей одной доле D_i=R_i:

$$S_{R_i} = \lceil \log_2 n \rceil n(p - 2^s)/p.$$

На примере, приведенном выше, видно, что несмотря на то, что из входной двоичной последовательности D данные считывались по s=3 бита (что позволяет получить значения от 0 до 7), в матрице R возникли значения, превосходящие 2^3, следовательно, хранить элементы долей, используя их трехбитное представление, невозможно

3. Метод ТХП и поля Галуа характеристики 2. Анализ метода ТХП показал, что он может быть реализован над полями Галуа характеристики 2, т. е. полей Галуа мощности 2^r, где $r(> 1) \in N$. Использование таких полей позволяет избежать описанных выше проблем.

Арифметика полей Галуа непростой мощности сложнее случаев простых полей, однако подобные вычисления используются довольно часто, например в помехоустойчивых кода Рида-Соломона, в стандартах шифрования, например *AES*. В работах [6, 7] приведены примеры реализации быстрых вычислений в некоторых полях Галуа. Для быстрых вычислений удобно использовать одновременно несколько представлений элементов поля F_{2^m} , а именно аддитивное в виде двоичного вектора длины m и мультипликативное (десятичный номер) в виде целого числа в диапазоне от 0 до $2^m - 1$ [8]. Сложение и вычитание происходят с использованием двоичных векторов по правилам поля F_2, а умножение и деление — с помощью десятичных номеров по формулам:

$$a * b = \begin{cases} 0, \text{если } a = 0 \text{ или } b = 0, \\ a + b - 1, \text{если } a \neq 0, b \neq 0 \text{ и } a + b \leq q, \\ a + b - q, \text{если } a \neq 0, b \neq 0 \text{ и } a + b > q, \end{cases} \quad (3)$$

$$a : b = \begin{cases} 0, \text{если } a = 0, b \neq 0, \\ a - b + 1, \text{если } b \neq 0, a \geq b, \\ q + a - b, \text{если } b \neq 0, a < b, \end{cases} \quad (4)$$

где $q = 2^m$— порядок поля.

Рассмотрим пример вычисления в поле Галуа $F_{2^3} = F_8$. В таблице 1 показаны различные представления элементов поля F_8 [9].

Таблица 1

Представление поля GF^3

Мультипликативное представление		Аддитивное представление	
Десятичный номер N	Степень α^i	Двоичный вектор $\alpha^2\alpha^1\alpha^0$	Полином $\sum_{i=0}^{i=r-1} \alpha^i x^i$
0	$\alpha^{-\infty}$	000	0
1	α^0	001	1
2	α^1	010	x
3	α^2	100	x^2
4	α^3	011	$x + 1$
5	α^4	110	$x^2 + x$
6	α^5	111	$x^2 + x + 1$
7	α^6	101	$x^2 + 1$

Пусть элементы поля заданы с помощью десятичных номеров a=3, b=5, вычислим a+b, a*b, a/b. Заметим, что из-за особенностей вычислений в двоичном поле a+b=a-b=b-a.

Для вычисления суммы a+b можно использовать произвольное аддитивное представление элементов поля. Например, используем полиномиальное представление элементов $a = x + 1$, $b = x^2 + 1$; сумма $(x + 1) + (x^2 + 1) = x^2 + x$ соответствует десятичному номеру 5; используем векторное представление элементов a=011_2, b=101_2; сумма 011_2+$101_2 = 110_2$ соответствует десятичному номеру 5.

Для вычисления произведения и деления удобно применять мультипликативное представление элементов поля. При использовании десятичных номеров следует использовать формулы (3) и (4), тогда 3*5=3+5–1=7, 3/5=8+3–5=6; при вычислениях с использованием степенного представления получаем a $\rightarrow \alpha^2$, $b \rightarrow \alpha^4$, $\alpha^2 * \alpha^4 = \alpha^{2+4} = \alpha^6 \rightarrow 7$, $\frac{\alpha^2}{\alpha^4} = \alpha^{2-4} = \alpha^{-2 \, mod \, 7} = \alpha^5 =$ $\rightarrow 6$.

4. Программное средство, реализующее метод ТХП. Для проведения экспериментов по оцениваю различных параметров изучаемого метода порогового разделения данных создано программное средство (ПС), реализующее метод ТХП, работающий с полями Галуа простой мощности и мощности 2^r, где целое $r > 1$.

Для программной реализации метода выбран язык программирования *C++*. Созданное ПС работает в двух режимах: разбиение исходного файла и восстановление исходного файла по набору долей. Вход ПС зависит от режима его работы. Для случая разбиения файла на вход ПС подаются

произвольные текстовые файлы, предназначенные для разбиения, а также параметры метода ТХП: *n*, *k* и мощность поля Галуа. Для случая восстановления файла на вход ПС подаются параметры метода и k частей разделенных данных.

Программа состоит из трех компилируемых файлов: *func.cpp* — где реализованы все функции программы, *main.cpp* — в нем находятся константы, переменные, в которые записываются результаты функций из *func.cpp*, и соответственно вызовы функций, а также *Header.h* — заголовочный файл с именами функций.

5. Эксперименты. В проведенных экспериментах не анализировалась зависимость времени работы программы в случае различных полей Галуа, так как полученный результат в большей мере зависел бы от качества написания программного кода, чем от особенностей метода.

В таблице 2 приведены результаты некоторых экспериментов, оценивающих избыточность метода в зависимости от мощности применяемых полей Галуа. Для проведения экспериментов, описанных в этой таблице, использовались два исходных файла, подлежащий разделению, их размеры соответственно 95 232 и 952 320 бит. Установлены параметры метода: *n*=17, *k*=8. Мощность поля Галуа и длина *s* отрезка бит, которые используются для получения одного элемента поля из двоичных входных данных, указаны в первом столбце таблицы. Во втором столбце указан размер входных данных. В третьем столбце в битах указан размер доли $D_i = R_i$, $i = \overline{1,n}$, отметим, что для зафиксированных параметров метода и размера входного файла размер доли является величиной постоянной. Суммарный размер всех долей в битах находится в четвертом столбце. Пятый столбец содержит размер вспомогательной структуры L_{R_i} или L_M, если параметры метода таковы, что ее создание необходимо, а также пометку о том, какая именно структура была построена (L_{R_i} или L_M). При создании таблицы полагаем, что вспомогательная структура хранится вместе с каждой долей, т. е. *n* раз, хотя выше было отмечено, что есть случаи, когда можно сократить число хранимых экземпляров L_M (см. раздел 2). Шестой столбец таблицы содержит суммарный размер всех хранимых вспомогательных структур. В седьмой столбец записан общий объем памяти, необходимый для хранения всех элементов разделенных данных.

Таблица 2

Размеры долей и вспомогательных структур метода ТХП

F_q, s	Размер входных данных, бит	Размер доли, бит	Суммарный размер всех долей R, бит	Размер вспомогательной структуры, бит	Суммарный размер всех вспомогательных структур, бит	Общий объем всех долей и вспомогательных структур, бит
1	2	3	4	5	6	7
F_{256}, s=8	95 232	11 968	203 456	0	0	203 456
F_{263}, s=8		11 968	203 456	704 (L_{R_i})	11 968	215 424
F_{251}, s=8		11 968	203 456	2 112 (L_M)	35 904	239 360
F_{251}, s=7		12 019	204 323	6 464 (L_{R_i})	109 888	314 211
F_{251}, s=7, s_R=8		13 736	233 512	0	0	233 512
F_{32}, s=5		11 985	203 745	0	0	203 745
F_{31}, s=5		11 985	203 745	4 512 (L_M)	76 704	280 449
F_{31}, s=4		11 968	203 456	11 264 (L_{R_i})	191 488	394 944

F_q, s	Размер входных данных, бит	Размер доли, бит	Суммарный размер всех долей R, бит	Размер вспомогательной структуры, бит	Суммарный размер всех вспомогательных структур, бит	Общий объем всех долей и вспомогательных структур, бит
F_{31}, s=4, s_R=5		14 960	254 320	0	0	254 320
F_{64}, s=6		11 934	202 878	0	0	202 878
F_{61}, s=6		11 934	202 878	5 616 (L_M)	95 472	298 350
F_{67}, s=6		11 934	202 878	936 (L_{R_i})	15 912	218 790
F_{67}, s=6, s_R=7		13 923	236 691	0	0	236 691
F_{1024}, s=10	952320	13 600	2 312 000	0	0	2 312 000
F_{1021}, s=10		13 600	2 312 000	43 520 (L_M)	739 840	3 051 840
F_{1031}, s=10		13 600	2 312 000	64 (L_{R_i})	10 880	2 322 880
F_{1031}, s=10, s_R=11		14 960	2 543 200	0	0	2 543 200

Таблица 3 построена по результатам таблицы 2 и содержит разницу в объемах занимаемой памяти между случаем поля характеристики 2 и простым полем: в первом и втором столбцах таблицы указаны поля Галуа и число бит, используемое для хранения элементов поля. Различие между столбцами состоит в том, что первый столбец связан с полями Галуа характеристики 2, а второй — с полями простой нечетной мощности. Третий столбец показывает, на сколько общий объем хранимых данных в случае поля нечетной простой мощности превосходит аналогичный объем для случая использования полей Галуа характеристики 2.

Таблица 3

Разница в общих объемах хранимых данных между случаем поля характеристики 2 и простым полем

Используемое поле и число бит для хранения элемента поля	Размер простого поля	Разница в объемах хранимых данных, %
F_{256}, s=8	F_{263}, s=8	5.9%
	F_{251}, s=7, s_R=8	14.7%
	F_{251}, s=8	17.7%
	F_{251}, s=7	54.4%
F_{32}, s=5	F_{31}, s=4, s_R=5	24.8%
	F_{31}, s=5	37.6%
	F_{31}, s=4	93.8%
F_{64}, s=6	F_{67}, s=6	7.8%
	F_{67}, s=6, s_R=7	16.6%
	F_{61}, s=6	47%
F_{1024}, s=10	F_{1031}, s=10	0.5%
	F_{1031}, s=10, s_R=11	10%
	F_{1021}, s=10	32%

Результаты проведенных экспериментов позволяют сделать ряд выводов:

1. При использовании полей характеристики 2 в методе ТХП объем хранимых долей минимальный.

2. При использовании полей простой нечетной мощности p с точки зрения объема хранимых данных предпочтительнее использование случаев, когда $2^s < p$ и $2^s \approx p$.

3. В случае хранения структуры L_M в простых полях, по сравнению с полями характеристики 2, рост общего объема хранимых данных составляет от 17 до 47 %.

4. В случае хранения структуры L_R в простых полях, по сравнению с полями характеристики 2, рост общего объема хранимых данных составляет от 0,5 до 94 %.

5. В случае, когда для хранения данных в долях используется большее число бит, чем при считывании, рост общего объема хранимых данных составляет от 14 до 25 %.

Заключение. В работе рассмотрен метод ТХП порогового разделения данных, авторы которого предлагают использовать в нем поля Галуа простой мощности. Однако в таком случае возникают дополнительные структуры данных, которые необходимо хранить и обрабатывать. В работе оценены размеры дополнительных структур, а также показано, что применение метода ТХП над полями Галуа характеристики 2 сокращает объем хранимых данных.

Библиографический список

1. Тормасов, А. Г. Модель распределенного хранения данных с регулируемой избыточностью / А. Г. Тормасов, М. А. Хасин, Ю. И. Пахомов // Исследовано в России : [сайт]. — 2001 — № 1–4 — С. 355–364. — URL: http://zhurnal.ape.relarn.ru/articles/2001/035.pdf (дата обращения: 11.11.2022).

2. The Development of Probabilistic Algorithm of Monitoring A Result Correctness for Cloud Computing in Residue Number System / N. I. Chervyakov, M. G. Babenko, A. S. Nazarov [et al.] // Engineering & Telecommunication – En&T-2015: Proceedings of International Conference. — Moscow, Russia: IEEE, 2015. — P. 196–198. DOI:10.1109/EnT.2015.12

3. Деундяк, В. М. Схема разделенной передачи конфиденциальных данных на основе дифференцирования полиномов нескольких переменных над простыми полями Галуа / В. М. Деундяк, Н. С. Могилевская // Вопросы кибербезопасности. — 2017. — № 5 (24). — С. 64–71. DOI: 10.21681/2311-3456-2017-5-64-71

4. Могилевская, Н. С. Пороговое разделение файлов на основе битовых масок: идея и возможное применение / Н. С. Могилевская, Р. В. Кульбикаян, Л. А. Журавлёв // Вестник Донского государственного технического университета : [сайт]. — 2011. — Т. 11, № 10. — С. 1749–1755. — URL: https://www.vestnik-donstu.ru/jour/article/view/912/907 (дата обращения: 27.10.2022).

5. Шафаревич, И. Р. Линейная алгебра и геометрия : учебное пособие для студентов высших учебных заведений / И. Р. Шафаревич, А. О. Ремизов. — Москва : ФИЗМАТЛИТ, 2009. — 512 с.

6. Рахман, П. А. Арифметика двоичного поля Галуа на базе быстрого умножения и инвертирования элементов поля и ее аппаратная реализация / П. А. Рахман // Международный журнал прикладных и фундаментальных исследований : [сайт]. — 2015. — № 12–3. — С. 403–408. — URL: https://applied-research.ru/ru/article/view?id=7942 (дата обращения: 15.12.2022).

7. Листопад, Е. В. Анализ подходов к реализации на FPGA операций умножения в поле Галуа / Е. В. Листопад // Доклады БГУИР : [сайт]. — 2018. — № 3 (113). — С. 5–12. — URL: https://cyberleninka.ru/article/n/analiz-podhodov-k-realizatsii-na-fpga-operatsiy-umnozheniya-v-pole-galua (дата обращения: 15.12.2022).

8. Муттер, В. М. Основы помехоустойчивой телепередачи информации / В. М. Муттер. — Ленинград : Энергоатомиздат, 1990. — 282 с.

Об авторах:

Могилевская Надежда Сергеевна, доцент кафедры «Алгебра и дискретная математика» Института механики, математики и компьютерных наук имени И. И. Воровича Южного федерального университета (344000, РФ, г. Ростов-на-Дону, ул. Мильчакова, 8 а), кандидат технических наук, nmogilevskaya@sfedu.ru

Гурова Светлана Евгеньевна, студентка кафедры «Алгебра и дискретная математика» Института механики, математики и компьютерных наук имени И. И. Воровича Южного федерального университета (344000, РФ, г. Ростов-на-Дону, ул. Мильчакова, 8 а), gurova@sfedu.ru

About the Authors:

Nadezhda S. Mogilevskaya, associate professor of the Algebra and Discrete Mathematics Department, Institute of Mechanics, Mathematics and Computer Science named after Vorovich I.I., Southern Federal University (8A, Milchakov str., Rostov-on-Don, 344000, RF), Cand. Sci. (Eng.), nmogilevskaya@sfedu.ru

Svetlana E. Gurova, student of the Algebra and Discrete Mathematics Department, Institute of Mechanics, Mathematics and Computer Science named after Vorovich I.I., Southern Federal University (8A, Milchakov str., Rostov-on-Don, 344000, RF), gurova@sfedu.ru