Научная статья
УДК 519.6+004.43
doi: 10.34822/1999-7604-2022-4-77-90

ОБЗОР МЕТОДОВ И ОЦЕНКА СЛОЖНОСТИ АЛГОРИТМОВ ОПЕРАЦИЙ СРАВНЕНИЯ В МОДУЛЯРНОЙ АРИФМЕТИКЕ И ПЕРЕВОДА ИЗ МОДУЛЯРНОЙ СИСТЕМЫ В ПОЗИЦИОННУЮ СИСТЕМУ СЧИСЛЕНИЯ

Наталья Сергеевна Золотарева
Сургутский государственный университет, Сургут, Россия
zolotareva_ns@surgu.ru, https://orcid.org/0000-0001-9751-4232

Аннотация. Изложено математическое описание методов выполнения немодульных операций в модулярной арифметике: перевод из модулярной системы счисления в позиционную систему счисления и операции сравнения в модулярной системе счисления. Разработана программа на языке Python, моделирующая выполнение алгоритмов на электронно-вычислительной машине. Приведены примеры и представлены результаты работы алгоритмов. Выполнены оценки сложности алгоритмов для их сравнения и выявления оптимальных.

Ключевые слова: модулярная арифметика, модульные операции, немодульные операции, сравнение чисел, Китайская теорема об остатках, ранг числа, сложность алгоритмов, математическое моделирование

Для цитирования: *Золотарева Н. С. Обзор методов и оценка сложности алгоритмов операций сравнения в модулярной арифметике и перевода из модулярной системы в позиционную систему счисления // Вестник кибернетики. 2022. № 4 (48). С. 77–90. DOI 10.34822/1999-7604-2022-4-77-90.*

Original article

METHODS REVIEW AND COMPLEXITY ESTIMATION OF THE ALGORITHMS FOR COMPARISON OPERATIONS IN MODULAR ARITHMETIC AND TRANSFER OPERATIONS FROM A MODULAR NUMBER SYSTEM TO A POSITIONAL NUMBER SYSTEM

Natalya S. Zolotareva
Surgut State University, Surgut, Russia
zolotareva_ns@surgu.ru, https://orcid.org/0000-0001-9751-4232

Abstract. The article presents a mathematical description of methods for performing non-modular operations in modular arithmetic, such as transfer operations from a modular number system to a positional number system and comparison operations in a modular number system. A program for simulating algorithms on a computer was developed in Python. The study provides examples and results of the algorithms' performance. The algorithms' complexity is estimated in order to compare them and determine the most effective one.

Keywords: modular arithmetic, modular operations, non-modular operations, number comparison, Chinese remainder theorem, number rank, complexity of algorithms, mathematical modeling

For citation: *Zolotareva N. S. Methods Review and Complexity Estimation of the Algorithms for Comparison Operations in Modular Arithmetic and Transfer Operations from a Modular Number System to a Positional Number System // Proceedings in Cybernetics. 2022. No. 4 (48). P. 77–90. DOI 10.34822/1999-7604-2022-4-77-90.*

ВВЕДЕНИЕ

Существует метод выполнения арифметических операций над большими целыми числами, который основан на положениях теории чисел. Идея этого метода состоит в том, чтобы оперировать не непосредственно чис-

лом A, а его остатками или вычетами $\alpha_1 = A(\mathrm{mod}\, p_1)$, $\alpha_2 = A(\mathrm{mod}\, p_2)$, …, $\alpha_n = A(\mathrm{mod}\, p_n)$, где p_1, p_2, …, p_n – модули, не содержащие общих делителей (взаимно-простых). Множество чисел, над которыми можно выполнять операции модулярной арифметики, – это $P = p_1 \cdot p_2 \cdot ... \cdot p_n$ (произведение модулей), вычеты определяются как $\alpha_i = A - \left[\frac{A}{p_i}\right] p_i, i = 1, 2, ..., n, \left[\frac{A}{p_i}\right]$ – целая часть меньшая или равная числу. Любое целое положительное число A из диапазона P можно представить в виде набора остатков от деления этого числа на выбранные основания системы, то есть $A = (\alpha_1, \alpha_2, ..., \alpha_n)$ [1].

Рассматривая вычисления в модулярной арифметике, все операции можно разбить на две группы: модульные и немодульные.

Операции сложения, вычитания и умножения в модулярной арифметике относятся к модульным. Это операции, в которых действия над числами можно проводить независимо в параллельных каналах [2]. На компьютере с параллельным выполнением операций применение модулярной арифметики дает значительное преимущество в скорости для модульных операций. Операции, связанные с разными модулями, могут выполняться одновременно, и возникает повышение скорости их выполнения.

Преимущество представления чисел в модулярной системе счисления заключается в том, что операции сложения, вычитания и умножения выполняются достаточно просто:

$$\begin{aligned}
&(\alpha_1, \alpha_2, ..., \alpha_n) + (\beta_1, \beta_2, ..., \beta_n) = \\
&= ((\alpha_1 + \beta_1) \bmod p_1, ..., (\alpha_n + \beta_n) \bmod p_n), \\
&(\alpha_1, \alpha_2, ..., \alpha_n) - (\beta_1, \beta_2, ..., \beta_n) = \\
&= ((\alpha_1 - \beta_1) \bmod p_1, ..., (\alpha_n - \beta_n) \bmod p_n), \\
&(\alpha_1, \alpha_2, ..., \alpha_n) \cdot (\beta_1, \beta_2, ..., \beta_n) = \\
&= ((\alpha_1 \cdot \beta_1) \bmod p_1, ..., (\alpha_n \cdot \beta_n) \bmod p_n).
\end{aligned} \quad (1)$$

Если при выполнении операции вычитания получен результат положительный, то операция выполняется вычитанием соответствующих цифр разряда, а если результат отрицательный, то берется ее дополнение к основанию [1].

Основной недостаток представления чисел в модулярной системе счисления состоит в том, что непросто проверить, является ли $(\alpha_1, \alpha_2, ..., \alpha_n)$ большим, чем $(\beta_1, \beta_2, ..., \beta_n)$. Трудно установить возникновение переполнения в результате выполнения операций сложения, вычитания и умножения, сложно выполнять операцию деления. Возникает необходимость выполнения немодульных операций, которые являются медленными, что вызывает определенные трудности.

Немодульные операции – это операции, в которых необходимо знать информацию обо всем числе и приходится восстанавливать позиционное представление числа. К немодульным операциям модулярной арифметики можно отнести следующие: перевод числа из модулярной системы счисления в позиционную, деление, вычисление позиционных характеристик (след, ядро, ранг и др.), расширение системы оснований, сравнение чисел, масштабирование чисел и др. [2–6].

МАТЕРИАЛЫ И МЕТОДЫ

К настоящему времени разработаны и продолжают разрабатывать алгоритмы выполнения перечисленных операций. Однако, несмотря на многочисленное количество работ в этой области [3, 6–12], еще остается много нерешенных вопросов.

В работе сосредоточено внимание на особенностях выполнения сложных и неочевидных алгоритмов. На первом этапе осуществлен анализ различных методов и алгоритмов выполнения модульных и немодульных операций в модулярной системе счисления. Далее в работе излагается материал относительно следующих немодульных операций: перевод из модулярной системы счисления (МСС) в позиционную систему счисления (ПСС) и сравнение чисел в МСС. Для каждой операции было выбрано два метода. Приведены их математическое описание и примеры расчета. На втором этапе были разработаны программы на языке Python, моделирующие выполнение алгоритмов на ЭВМ. На третьем этапе произведено сравнение алгоритмов между собой, а именно выполнена оценка сложности алгоритмов с целью выявления оптимальных.

Рассмотрим анализ существующих методов выполнения немодульных операций.

1. Перевод из модулярной системы счисления в позиционную систему счисления.

Переход из модулярной системы счисления в позиционную систему счисления выполняется с основанием, равным 10. То есть целое положительное число A представимо в виде:

$$A = a_n \cdot 10^n + a_{n-1} \cdot 10^{n-1} + ... + a_1 \cdot 10^1 + a_0,$$

где a_i – целые десятичные цифры в интервале [0, 9].

Операцию перевода чисел из модулярной системы счисления в позиционную можно считать одной из основных немодульных операций, так как определение величины числа необходимо для других немодульных операций: сравнение чисел по величине, определение знака и др. Эффективность выполнения этих операций будет напрямую зависеть от того, насколько эффективным будет метод определения величины числа.

Существуют как традиционные (метод ортогональных базисов, метод перевода в обобщенную позиционную систему счисления), так и новые методы перевода (интервальные методы перевода) [3, 6, 7].

Метод ортогональных базисов.

Основой метода ортогональных базисов является Китайская теорема об остатках (КТО) [7].

Теорема: Пусть $p_1, p_2, \ldots, p_n$ – попарно взаимно простые числа, $P = p_1 \cdot p_2 \cdot ... \cdot p_n$, $y_1, y_2, ..., y_n$ подобраны так, что $\frac{P}{p_i} y_i \equiv \equiv 1 (\mathrm{mod}\ p_i)$, $A_0 = \Sigma_{i=1}^{n} \frac{P}{p} y_i \alpha_i$, $i = 1, 2, ..., n$. Тогда решение системы $A = \alpha_i (\mathrm{mod}\ p_i)$, $i = 1, 2, ..., n$ будет иметь вид:

$$A \equiv A_0 (\mathrm{mod}\ P). \qquad (2)$$

Пусть основания модулярной системы $p_1, p_2, \ldots, p_n$, $P = p_1 \cdot p_2 \cdot ... \cdot p_n$ – мощность диапазона модулярной системы. С выбором системы определяются ее основные константы – базисы $B_i = (\beta_{1i}, \beta_{2i}, ..., \beta_{ni})$, $i = 1, 2, \ldots, n$. Задача перевода числа $A = (\alpha_1, \alpha_2, ..., \alpha_n)$ в ПСС заключается в определении таких чисел M_i, $i = 1, 2, \ldots, n$, чтобы $A = \Sigma_{i=1}^{n} M_i B_i$. Для однозначного определения M_i на базисы системы B_i накладывается ряд ограничений и показывается, что таким свойством обладают базисы [13]:

$$\begin{aligned} B_1 &= (1, 0, 0, ..., 0, 0), \\ B_2 &= (0, 1, 0, ..., 0, 0), \\ &\ldots \\ B_n &= (0, 0, 0, ..., 0, 1), \end{aligned} \qquad (3)$$

которые называют ортогональными.

Тогда в случае ортогональных базисов $M_i = \alpha_i, i = 1, 2, ..., n$. Ортогональные базисы определяют по формуле:

$$B_i = \frac{m_i \cdot P}{p_i} = m_i \cdot P_i, i = 1, 2, ..., n, \qquad (4)$$

где $P_i = \frac{P}{p_i}$,

m_i – целые положительные числа, которые называются весами базиса. Их определяют из сравнений:

$$P_i \cdot m_i \equiv 1 (\mathrm{mod}\ p_i). \qquad (5)$$

Тогда, по КТО, число:

$$A = (\alpha_1, \alpha_2, ..., \alpha_n) \equiv \Sigma_{i=1}^{n} \alpha_i B_i (\mathrm{mod}\ P). \qquad (6)$$

Таким образом, если найдены ортогональные базисы для системы оснований, то для перевода числа $A = (\alpha_1, \alpha_2, ..., \alpha_n)$ достаточно вычислить $\Sigma_{i=1}^{n} \alpha_i B_i$ и ввести эту сумму в диапазон [0; P) вычитанием величины, кратной P, т. е.:

$$A = | \Sigma_{i=1}^{n} \alpha_i B_i |_P = \Sigma_{i=1}^{n} \alpha_i B_i - r_A P, \qquad (7)$$

где r_A – ранг числа A, показывающий, сколько раз надо вычесть мощность диапазона P из полученного числа, чтобы вернуть его в диапазон [14].

Пример 1. Пусть дана система оснований $p_1 = 32765$, $p_2 = 32767$, $p_3 = 32768$, $p_4 = 32769$, $p_5 = 32771$. Мощность диапазона модулярной системы $P = p_1 \cdot p_2 \cdot p_3 \cdot p_4 \cdot p_5 =$

$= 377789315111134411161 60$. Переведем число, представленное в МСС A = (36, 4, 0, 4, 36), в ПСС.

Вычислим ортогональные базисы. Для этого найдем величин P_i по формуле:

$$P_i = \frac{P}{p_i}, \tag{8}$$

где P – диапазон модулярной системы, p_1, p_2, …, p_n – основания.

$$P_1 = \frac{P}{p_1} = \frac{377789315111134411161 60}{32765} = 1153027056649273344,$$

$$P_2 = \frac{P}{p_2} = \frac{377789315111134411161 60}{32765} = 1152956679314964480,$$

$$P_3 = \frac{P}{p_3} = \frac{377789315111134411161 60}{32765} = 1152921493869428745,$$

$$P_4 = \frac{P}{p_4} = \frac{377789315111134411161 60}{32765} = 1152886310571376640,$$

$$P_5 = \frac{P}{p_5} = \frac{377789315111134411161 60}{32765} = 1152815950416936960.$$

Веса базисов вычислим по формуле (5).

Для нашего случая:

$$1152921493869487 45 \cdot m_3 \equiv 1(\mathrm{mod}\, 32768).$$

Вычислим m_3 методом цепных дробей, содержащим этапы:

$$\begin{gathered} P_3 \cdot m_3 \equiv 1(\mathrm{mod}\, p_3) \\ m_3 = (-1)^n \cdot F_{n-1} \cdot 1(\mathrm{mod}\, p_3), \end{gathered} \tag{9}$$

где $F_0 = q_0, F_1 = q_0 \cdot q_1 + 1, ..., F_n = q_n \cdot F_{n-1} + F_{n-2}$.

$$\frac{p_3}{P_3} = \frac{32768}{1152921493869428845} = \left[q_0, q_1, q_2, q_3, q_4\right] = \left[0, 35184371761152, 36340, 1, 8\right],$$

$$\frac{1}{35184371761152 + \dfrac{1}{3640 + \dfrac{1}{1 + \dfrac{1}{8}}}}.$$

Последним элементом является q_4, и индекс равен 4, следовательно, n = 4.

Получим:

$$\begin{aligned} m_3 &= (-1)^n \cdot F_{n-1} \cdot 1(\mathrm{mod}\, p_1) \\ &= (-1)^4 \cdot F_{4-1} \cdot 1(\mathrm{mod}\, 32768) \\ &= 1 \cdot F_3 \cdot 1(\mathrm{mod}\, 32768) \\ &= 1 \cdot 3641 \cdot 1(\mathrm{mod}\, 32768) \\ &= 3641 \end{aligned}$$

$$F_3 = q_3 \cdot F_{3-1} + F_{3-2} = q_3 \cdot F_2 + F_1 =$$

$$= q_3(q_2(q_0 \cdot q_1 + 1) + q_0) + q_0 \cdot q_1 + 1 =$$

$$= 1(3640(0 \cdot 35184371761152 + 1) + 0) +$$

$$+ 0 \cdot 35184371761152 + 1 = 3641$$

$$1152921493869428745 \cdot m_3 = 3641(\mathrm{mod}\, 32768),$$

$$m_3 = 3641(\mathrm{mod}\, 32768) = 3641.$$

Аналогично вычисляются константы m_1, m_2, m_4, m_5:

m_1 = 9784, m_2 = 30719, m_4 = 2048, m_5 = 1934.

Определим ортогональные базисы по формуле (4):

$$B_1 = m_1 \cdot P_1 = 9784 \cdot 1153027056649273344 = 11281216722256490397696,$$

$$B_2 = m_2 \cdot P_2 = 30719 \cdot 1152956679314964480 = 35417676231876393861120,$$

$$B_3 = m_3 \cdot P_3 = 3641 \cdot 1152921493869428745 = 4197787159178590060545,$$

$$B_4 = m_4 \cdot P_4 = 9784 \cdot 1152886310571376640 = 2361111164050179358720,$$

$$B_5 = m_5 \cdot P_5 = 19344 \cdot 1152815950416936960 = 22300071744865228554240.$$

Вычислим величину числа A согласно формуле (6), зная, что $A = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ = = (36, 4, 0, 4, 36):

$$A = (\alpha_1 \cdot B_1 + \alpha_2 \cdot B_2 + \alpha_3 \cdot B_3 + \alpha_4 \cdot B_4 + \alpha_5 \cdot B_5) \times$$
$$\times (\mathrm{mod}\, P) = (36 \cdot 11281216722256490397696 +$$
$$+ 4 \cdot 35417676231876393861120 +$$
$$+ 0 \cdot 4197787159178590060545 +$$
$$+ 4 \cdot 2361111164050179358720 +$$
$$+ 36 \cdot 22300071744865228554240) \times$$
$$\times (\mathrm{mod}\, 3778931511113441116160) =$$
$$= (4061238020012336543170 56 +$$
$$+ 1416707049275055754444 80 + 0 +$$
$$+ 9444444656200717434880 +$$
$$+ 80280258281514822795 2640) \times$$
$$\times (\mathrm{mod}\, 3778931511113441116160) =$$
$$= 13600415344000881751490 56 \times$$
$$\times (\mathrm{mod}\, 3778931511113441116160) =$$
$$= 4294967296.$$

Получаем A = 4294967296.

Метод перевода определения величины числа из МСС в обобщенную позиционную систему счисления (ОПСС).

Пусть МСС задается основаниями p_1, p_2, …, p_n, и A = (α_1, α_2, …, α_n) – число в этой системе. Число в ОПСС задается кортежем $[x_1, x_2, ..., x_n]$, основаниями системы являются следующие константы p_1, p_1p_2, $p_1p_2p_3$, …, $p_1p_2 \ldots p_{n-1}$. Число X в ОПСС можно представить в виде:

$$X = x_n p_1 p_2 \ldots p_{n-1} + x_{n-1} p_1 p_2 \ldots p_{n-2} + \\ + \ldots + x_3 p_1 p_2 + x_2 p_1 + x_1, \qquad (10)$$

где $0 \le x_k \le \Pi_{i-1}^{k-1} p_i$, i = 1, 2, …, n – коэффициенты ОПСС [6].

Очевидно, что диапазоны чисел, представимых в МСС и ОПСС, совпадают, имеется взаимно-однозначное соответствие между множеством представлений чисел в МСС и ОПСС.

Равенство (10) можно переписать в виде:

$$X = x_1 + p_1(x_2 + p_2(x_3 + \ldots + \\ + p_{n-2}(x_{n-1} + p_{n-1}x_n) \ldots)).$$

Откуда следует, что цифры ОПСС могут быть получены из соотношений:

$$x_1 = X - \left[\frac{x}{p_1}\right] p_1 = X - X_1 p_1,$$

где $X_1 = \left[\frac{x}{p_1}\right]$,

$$x_2 = X_1 - \left[\frac{x_1}{p_2}\right] p_2 = X_1 - X_2 p_2,$$

где $X_2 = \left[\frac{x_1}{p_2}\right]$, (11)

…

$$x_n = X_{n-1} - \left[\frac{x_{n-1}}{p_n}\right] p_n = X_{n-1} - X_n p_n,$$

где $X_n = \left[\frac{x_{n-1}}{p_n}\right]$.

При определении цифр x_i по формулам (11) все вычисления можно вести в МСС.

Из (11) следует, что $x_i = |X|_{p_1}$, то есть x_1 – первая МСС цифра, или $x_1 = \alpha_1$. Для получения x_1 сперва $X - x_1$ представим в остаточной коде. Очевидно, $X - x_1$ делится на p_1. Более того, p_1 взаимно просто со всеми другими модулями. Следовательно, для нахождения цифры x_2 может быть использована процедура деления без остатка: $x_2 = \left|\frac{X - x_1}{p_1}\right|_{p_2}$.

Таким путем с помощью вычитаний и делений в остаточной записи все цифры ОПСС могут быть получены. При этом

$$x_1 = |X|_{p_1}, \; x_2 = \left|\left[\frac{X}{p_1}\right]\right|_{p_2}, \; x_3 = \left|\left[\frac{X}{p_1 p_2}\right]\right|_{p_3}, \; \ldots,$$
$$x_i = \left|\left[\frac{X}{p_1 p_2 \ldots p_{i-1}}\right]\right|_{p_3}, \; i > 1.$$

Перевод, осуществляемый согласно алгоритму (11), содержит всего $\frac{n(n-1)}{2}$ остаточные арифметические операции вычитания и деления без остатка, где n – число модулей системы. Может быть предложена некоторая модификация рассмотренного алгоритма в том плане, что операция деления заменяется

операцией умножения. Для этого предварительно вычисляется $\frac{n(n-1)}{2}$ констант t_{kj}, которые удовлетворяют условию:

$$t_{kj} p_k \equiv 1(mod\ p_j),\ (1 \le k < j \le n).$$

Эти константы можно получить, например, из расширенного алгоритма Евклида:

$$t_{kj} p_k + \gamma\, p_j = НОД(p_k, p_j) = 1 \text{ [15]}.$$

Здесь следует заметить тот факт, что константы t_{kj} полностью определяются к выбранной системой основания, поэтому могут быть вычислены заранее и храниться в некоторой таблице.

Если константы t_{kj} вычислены, то вычисление цифр x_i ОПСС по алгоритму (11) может быть переписано в виде [6]:

$$\begin{aligned} x_1 &\equiv \alpha_1 (mod\ p_1), \\ x_2 &\equiv (\alpha_2 - x_1)\cdot t_{12} (mod\ p_2), \\ x_3 &\equiv ((\alpha_3 - x_1)\cdot t_{13} - x_2)\cdot t_{23} (mod\ p_3), \\ &\vdots \\ x_n &\equiv ((\ldots(\alpha_n - x_1)\cdot t_{1n} - x_2) - \ldots - x_{n-1})\times \\ &\quad \times t_{n-1n} (mod\ p_n). \end{aligned} \qquad (12)$$

Константы t_{kj} принято также записывать в виде:

$$t_{kj} = \left| \frac{1}{p_k} \right|_{p_j} \qquad (13)$$

и называть обратными элементами по умножению для чисел p_k по модулю p_j.

Пример 2. Пусть дана система оснований $p_1 = 32765,\ p_2 = 32767,\ p_3 = 32768,\ p_4 = 32769,\ p_5 = 32771$. Мощность диапазона модулярной системы $P = p_1 \cdot p_2 \cdot p_3 \cdot p_4 \cdot p_5 = 377789315111113441116160$. Переведем число, представленное в МСС $A = (36, 4, 0, 4, 36)$, в ПСС.

Сначала найдем константы t_{kj}, используя формулу (13):

$$t_{12} = \left| \frac{1}{p_1} \right|_{p_2} = \left| \frac{1}{32765} \right|_{32767} = 16383.$$

Опишем, как вычислить $\left| \frac{1}{32765} \right|_{32767}$.

Эта запись равносильна $\frac{1}{32765} (mod\ 32767)$.

Введем следующее обозначение:

$$\frac{1}{p_1} = x,$$
$$p_1 \cdot x \equiv 1 (mod\ p_2).$$

То есть $p_1 \cdot x - 1 \vdots m$.

Тогда:

$$\frac{1}{32765} = x.$$

Вычислим сравнение:

$$32765 \cdot x \equiv 1 (mod\ 32767).$$

Здесь можно воспользоваться алгоритмом цепных дробей:

$$1 + \cfrac{1}{16382 + \cfrac{1}{2}}$$

$$\frac{32767}{32765} = [q_0, q_1, q_2] = [1, 16382, 2].$$

$n = 2$, так как q_2 последний элемент.

$$\begin{aligned} x &= (-1)^n \cdot F_{n-1} \cdot 1 (mod p_2) = \\ &= (-1)^2 \cdot F_{2-1} \cdot 1 (mod\ 32767) = \\ &= 1 \cdot F_1 \cdot 1 (mod\ 32767) = 16383 (mod\ 32767) = \\ &= 16383. \end{aligned}$$

$$F_1 = q_0 \cdot q_1 + 1 = 1 \cdot 16382 + 1 = 16383.$$

Вернемся к нашему сравнению: $32765 \cdot x \equiv \equiv 1 (mod\ 32767)$.

Подставим $x = 16383$:

$$32765 \cdot 16383 \equiv 1 (mod\ 32767),$$
$$536788995 \equiv 1 (mod\ 32767).$$

Действительно, $536788995 - 1 \vdots 32767$.
То есть:

$$t_{12} = \left| \frac{1}{p_1} \right|_{p_2} = \left| \frac{1}{32765} \right|_{32767} = 16383.$$

Далее аналогично находим остальные константы t_{kj}:

$$t_{12} = \left| \frac{1}{32765} \right|_{32767} = 16383;$$
$$t_{13} = \left| \frac{1}{32765} \right|_{32768} = 21845;$$
$$t_{14} = \left| \frac{1}{32765} \right|_{32769} = 8192;$$
$$t_{15} = \left| \frac{1}{32765} \right|_{32771} = 27309;$$
$$t_{23} = \left| \frac{1}{32767} \right|_{32768} = 32767;$$
$$t_{24} = \left| \frac{1}{32767} \right|_{32769} = 16384;$$
$$t_{25} = \left| \frac{1}{32767} \right|_{32771} = 24578;$$
$$t_{34} = \left| \frac{1}{32768} \right|_{32769} = 16384;$$
$$t_{35} = \left| \frac{1}{32768} \right|_{32771} = 21847;$$
$$t_{45} = \left| \frac{1}{32769} \right|_{32771} = 16385.$$

Константы t_{kj} запишем в виде матрицы $k \times j$:

$$\begin{pmatrix} 0 & 16383 & 21845 & 8192 & 27309 \\ 0 & 0 & 32767 & 16384 & 24578 \\ 0 & 0 & 0 & 16384 & 21847 \\ 0 & 0 & 0 & 0 & 16385 \end{pmatrix}.$$

Далее найдем x_1, x_2, x_3, x_4 по формуле (12), зная, что:

$$A = (\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5) = (36, 4, 0, 4, 36);$$
$$p_1 = 32765,\ p_2 = 32767, p_3 = 32768,$$
$$p_4 = 32769, p_5 = 32771,$$
$$x_1 \equiv \alpha_1 (mod\ p_1) = 36 (mod\ 32765) = 36,$$
$$x_2 \equiv (\alpha_2 - x_1) \cdot t_{12} (mod\ p_2) =$$
$$= (4 - 36) \cdot 16383 (mod\ 32767) =$$
$$= -524256 (mod\ 32767) = 16,$$
$$x_3 \equiv ((\alpha_3 - x_1) \cdot t_{13} - x_2) \cdot t_{23} (mod\ p_3) =$$
$$= ((0 - 36) \cdot 21845 - 16) \cdot 32767 (mod 32768) =$$
$$= -25769148412 (mod\ 32768) = 4,$$
$$x_4 \equiv (((\alpha_4 - x_1) \cdot t_{14} - x_2) \cdot t_{24}) - x_3) \cdot t_{34}) (mod\ p_4) =$$
$$= (((4 - 36) \cdot 8192 - 16) \cdot 16384 - 4) \cdot 16384) \times$$
$$\times (mod\ 32769) = -70373039210496 (mod\ 32769) = 0,$$
$$x_5 \equiv ((((\alpha_5 - x_1) \cdot t_{15} - x_2) \cdot t_{25}) - x_3) \cdot t_{35}) - x_4) \cdot t_{45}) \times$$
$$\times (mod\ p_5) = (((36 - 36) \cdot 27309 - 16) \cdot 24578 - 4) \times$$
$$\times 21847) - 0) \cdot 16385) (mod\ 32771) =$$
$$= -140769703034940 (mod\ 32771) = 0.$$

Запишем результат в виде табл. 1.

Таблица 1

Промежуточные вычисления метода перевода в обобщенную позиционную систему счисления

Операции	Модули					Цифры ОПСС
	$p_1 = 32765$	$p_1 = 32767$	$p_1 = 32768$	$p_1 = 32769$	$p_1 = 32771$	
A	36	4	0	4	36	$x_1 = 36$
$-$						
x_1	36	36	36	36	36	

Окончание табл. 1

Операции	Модули					Цифры ОПСС
	$p_1 = 32765$	$p_1 = 32767$	$p_1 = 32768$	$p_1 = 32769$	$p_1 = 32771$	
$A - x_1$		32735	32732	32737	0	
$\times$	0					
t_{1j}		16383	21845	8192	27309	
A_1		16	12	8	0	
$-$						$x_2 = 16$
x_2		16	16	16	16	
$A_1 - x_2$			32764	32761	32755	
$\times$		0				
t_{2j}			32767	16384	24578	
A_2			4	4	4	
$-$						$x_3 = 4$
x_3			4	4	4	
$A_2 - x_3$				0	0	
$\times$			0			
t_{3j}				16384	21847	
A_3				0	0	
$-$						$x_4 = 0$
x_4				0	0	
$A_3 - ax_4$					0	
$\times$				0		
t_{4j}					16385	
A_4					0	$x_5 = 0$

Примечание: составлено автором на основании данных, полученных в исследовании.

Применяя формулу (10), получим:

$$X = x_5\, p_1\, p_2\, p_3\, p_4 + x_4\, p_1\, p_2\, p_3 + x_3\, p_1\, p_2 + x_2\, p_1 + x_1 =$$
$$= 0 \cdot 32765 \cdot 32767 \cdot 32768 \cdot 32769 + 0 \cdot 32765 \cdot 32767 \times$$
$$\times 32768 + 4 \cdot 32765 \cdot 32767 + 16 \cdot 32765 + 36 =$$
$$= 0 + 0 + 4294443020 + 524240 + 36 = 42944967296.$$

2. Сравнение чисел в МСС.

Исследование и совершенствование существующих, а также разработка новых методов и алгоритмов реализации немодульной операции сравнения данных, представленных в модулярной арифметике, является важной и актуальной задачей [8–12]. Операция сравнения чисел широко используется при реализации большинства алгоритмов. Можно заметить, что в модулярной арифметике имеется три группы методов сравнения [8–10]. К первой группе относятся методы, основанные на преобразовании чисел из модулярной системы счисления в позиционную, далее выполнятся сравнение полученных чисел. Ко второй группе относятся методы, основанные на применении понятия нулевизации [3]. К третьей группе относятся методы, основанные на использовании позиционных характеристик.

Метод, основанный на переводе из чисел из МСС в ПСС.

Для перевода числа из МСС в ППС используется стандартное восстановление с помощью КТО [3], которую можно записать формулой:

$$X = \left| \sum_{i=1}^{n} P_i \cdot x_i \cdot \left| P_i^{-1} \right|_{p_i} \right|_P, \qquad (14)$$

где $P_i = \dfrac{P}{p_i}$, а $\left| P_i^{-1} \right|_{p_i}$ – мультипликативная инверсия P_i по модулю p_i.

Рассмотрим пример восстановления числа по формуле (14) и сравнения чисел.

Пример 3. Пусть задана МСС $p_1 = 32765,\ p_2 = 32767, p_3 = 32768, p_4 = 32769,$ и $p_5 = 32771$ числа $X = (36, 4, 0, 4, 36)$, $Y = (9, 1, 0, 1, 9)$. Мощность диапазона модулярной системы счисления равна $P = p_1 \cdot p_2 \cdot p_3 \cdot p_4 \cdot p_5 = 377789315111113441116160$. Сравним числа X и Y.

Применяя формулы (8) и (5), вычислим P_i, m_i.

Из примера 1 $P_1, P_2, P_3, P_4, P_5, m_1, m_2, m_3, m_4, m_5$, соответственно, равны:

$$P_1 = 1153027056649273344,$$
$$P_2 = 1152956679314964480,$$
$$P_3 = 1152921493869428745,$$
$$P_4 = 1152886310571376640,$$
$$P_5 = 1152815950416936960,$$
$$m_1 = 9784,\ \ m_2 = 30719,\ \ m_3 = 3641,$$
$$m_4 = 2048,\ \ m_5 = 19344.$$

Применяя формулу (14), найдем значение чисел X и Y:

$$X = \left| \begin{array}{l} 1153027056649273344 \cdot 36 \cdot 9784 + \\ +1152956679314964480 \cdot 4 \cdot 30719 + \\ +1152921493869428745 \cdot 0 \cdot 3641 + \\ +1152886310571376640 \cdot 4 \cdot 2048 + \\ +1152815950416936960 \cdot 36 \cdot 19344 \end{array} \right|_{377789315111113441116160} =$$
$$= \left| \begin{array}{l} 406123802001233654317056 + \\ +1416707049275055754444480 + \\ +9444444656200717434880 + \\ +8028025828151482279526 40 \end{array} \right|_{377789315111113441116160} =$$
$$= \left| 1360041534400088175149056 \right|_{377789315111113441116160} =$$
$$= 4294967296.$$
$$Y = \left| \begin{array}{l} 1153027056649273344 \cdot 9 \cdot 9784 + \\ +1152956679314964480 \cdot 1 \cdot 30719 + \\ +1152921493869428745 \cdot 0 \cdot 3641 + \\ +1152886310571376640 \cdot 1 \cdot 2048 + \\ +1152815950416936960 \cdot 9 \cdot 19344 \end{array} \right|_{377789315111113441116160} =$$
$$= \left| \begin{array}{l} 1015309505003084135792 64 + \\ +3541767623187639386112 0 + \\ +2361111164050179358720 + \\ +2007006457037870569881 60 \end{array} \right|_{377789315111113441116160} =$$
$$= \left| 3400103836000220437872 64 \right|_{377789315111113441116160} =$$
$$= 1073741824.$$

Так как $4294967296 > 1073741824$, значит $X > Y$.

Метод с использованием позиционных характеристик.

Отличным от вышеизложенного метода сравнения чисел является метод на основе использования минимальной функции ядра Акушского [10].

Была предложена аналитическая функция для вычисления *Pirlo* функции:

$$Pi(X) = \left| \sum_{i=1}^{n} k_i^{**} \cdot x_i \right|_{P_n}, \qquad (15)$$

где $k_i^{**} = \dfrac{\left| P_i^{-1} \right|_{p_i} \cdot P_i}{p_n}$.

Так как функция *Pirlo* является монотонно возрастающей, то она может быть использована для сравнения чисел, т. е. если $Pi(X) < Pi(Y)$, то $X < Y$. Однако возможны случаи, когда $Pi(X) = Pi(Y)$, и в этом случае $X < Y$, когда $x_n < y_n$.

Пример 4. Пусть задана МСС $p_1 = 32765,\ p_2 = 32767, p_3 = 32768, p_4 = 32769,$ $p_5 = 32771$ и числа $X = (36, 4, 0, 4, 36)$, $Y = (9, 1, 0, 1, 9)$. Мощность диапазона модулярной системы счисления равна $P = p_1 \cdot p_2 \cdot p_3 \cdot p_4 \cdot p_5 = 377789315111113441116160$. Сравним числа X и Y.

Вычислим значения:

$$P_5 = 1152815950416936960,$$
$$k_1^{**} = \frac{\left| P_1^{-1} \right|_{p_1} \cdot P_1}{p_5} = \frac{9784 \cdot 1153027056649273344}{32771} =$$
$$= 344243896196530176,$$
$$k_2^{**} = \frac{\left| P_2^{-1} \right|_{p_2} \cdot P_2}{p_5} = \frac{30719 \cdot 1152956679314964480}{32771} =$$
$$= 1080762754626846720,$$
$$k_3^{**} = \frac{\left| P_3^{-1} \right|_{p_3} \cdot P_3}{p_5} = \frac{3641 \cdot 1152921493869428745}{32771} =$$
$$= 128094570174196395,$$
$$k_4^{**} = \frac{\left| P_4^{-1} \right|_{p_4} \cdot P_4}{p_5} = \frac{2048 \cdot 1152886310571376640}{32771} =$$
$$= 72048798146232320,$$

$$k_5^{**} = \frac{\left|P_5^{-1}\right|_{p_5} \cdot P_5}{p_5} = \frac{19344 \cdot 1152815950416936960}{32771} =$$
$$= 680481881690068309.$$

Найдем значение функции $Pirlo$ для чисел X и Y по формуле (15):

$$Pi(X) = \begin{vmatrix} 344243896196530176 \cdot 36 + \\ +1080762754626846720 \cdot 4 + \\ +128094570174196395 \cdot 0 + \\ +72048798146232320 \cdot 4 + \\ +680481881690068309 \cdot 36 \end{vmatrix}_{1152815950416936960} =$$
$$= \begin{vmatrix} 12392780263075086336 + \\ +4323051018507386880 + \\ +288195192584929280 + \\ +24497347740842459124 \end{vmatrix}_{1152815950416936960} =$$
$$= \left|4150137421500986162 0\right|_{1152815950416936960} = 131060$$

$$Pi(Y) = \begin{vmatrix} 344243896196530176 \cdot 9 + \\ +1080762754626846720 \cdot 1 + \\ 128094570174196395 \cdot 0 + \\ +72048798146232320 \cdot 1 + \\ +680481881690068309 \cdot 9 \end{vmatrix}_{1152815950416936960} =$$
$$= \begin{vmatrix} 3098195065768771584 + \\ +1080762754626846720 + \\ +72048798146232320 + \\ 6124336935210614781 \end{vmatrix}_{1152815950416936960} =$$
$$= \left|10375343553752465405\right|_{1152815950416936960} = 32765$$

и поскольку $Pi(X) > Pi(Y)$, то $X > Y$.

РЕЗУЛЬТАТЫ И ИХ ОБСУЖДЕНИЕ

На втором этапе на высокоуровневом языке программирования Python были разработаны алгоритмы выполнения рассмотренных немодульных операций. Приведем фрагменты алгоритмов и результаты расчетов (рис. 1, 2).

```
Base = [32765, 32767, 32768, 32769, 32771]  M = 377789315111134411161 60
######### Перевод из модулярной системы счисления в позиционную систему счисления #########
######### Метод ортогональных базисов #########
(36, 4, 0, 4, 36)
4294967296
rank (простая версия) = 36
----------------- Определение ранга по алгоритму -----------------
M =  [[1, 1, 1, 1, 1], [0, 32765, 32765, 32765, 32765], [0, 0, 3, 8, 24], [0, 0, 0, 32761, 32699], [0, 0, 0, 0, 144]]
M_rank =  [2, 55746, 15, 21349, 85]
[0, 32733, 32729, 32733, 32765] 55674
[0, 0, 15, 40, 120] 39
[0, 0, 0, 32761, 32699] 21313
[0, 0, 0, 0, 144] 49
[0, 0, 0, 0, 0] -37
rank (по алгоритму) = 36
######### Метод перевода в обобщенную позиционную систему счисления #########
(36, 4, 0, 4, 36)
----------------- Перевод в ПСС (вариант 2) -----------------
t_matrix =  [[0, 16383, 21845, 8192, 27309], [0, 0, 32767, 16384, 24578], [0, 0, 0, 32768, 21847], [0, 0, 0, 0, 16385]]
A_matrix =  [36, 16, 4, 0, 0]
4294967296
```

Рис. 1. Результаты расчетов по алгоритму «Перевод из модулярной системы счисления в позиционную систему счисления»

Примечание: составлено автором на основании данных, полученных в исследовании.

```
######### Сравнение чисел в МСС #########
######### Метод, основанный на переводе из чисел из МСС в ПСС #########
Base = [32765, 32767, 32768, 32769, 32771]  M = 377789315111134411161 60
(36, 4, 0, 4, 36)  <  (9, 1, 0, 1, 9)
False

######### Метод с использованием позиционных характеристик #########
Base = [32765, 32767, 32768, 32769, 32771]  M = 377789315111134411161 60
(36, 4, 0, 4, 36) 4294967296
(9, 1, 0, 1, 9) 1073741824
----------------- сравнения чисел в СОК с использованием позиционных характеристик -----------------
k =  [344243896196530176, 1080762754626846720, 128094570174196395, 72048798146232320, 680481881690068309]
Pi (36, 4, 0, 4, 36)  =  131060
Pi (9, 1, 0, 1, 9)  =  32765
1
```

Рис. 2. Результаты расчетов по алгоритму «Сравнение чисел в МСС»

Примечание: составлено автором на основании данных, полученных в исследовании.

Сравнив математическое описание и результаты расчетов по разработанным алгоритмам, убеждаемся в совпадении результатов, что свидетельствует о корректности вычислений. Сравнение проводилось с применением процесса математического моделирования. Объектом моделирования являются методы выполнения операций перевода и сравнения чисел в МСС. В качестве математической модели выступает совокупность математических формул и отношений между ними, которая адекватно отражает концепцию рассматриваемых методов. Процесс математического моделирования можно представить в виде следующих этапов: исследование объекта моделирования, постановка задачи, анализ, выбор методов, поиск решения, разработка алгоритма решения, проверка, т. е. соответствие результатов математической модели и разработанных алгоритмов, практическое использование и анализ результатов моделирования.

Выполнена оценка сложности разработанных алгоритмов. При оценке вычислительной сложности алгоритмов используют два подхода: временная вычислительная сложность и асимптотическая вычислительная сложность.

Определение: вычислительная сложность (алгоритмическая сложность) – функция зависимости объема работы алгоритма от размера обрабатываемых данных.

Вычислительная сложность отвечает на центральный вопрос при разработке алгоритмов: как изменится время исполнения и объем занятой памяти в зависимости от размера входных данных.

Определение: временная сложность алгоритма – это функция от размера входных данных, равная количеству элементарных операций, выполняемых алгоритмом для решения экземпляра задачи указанного размера.

Вычислительная сложность – это количественная оценка ресурсов, затрачиваемых алгоритмом. Вычислительная сложность является более общим термином, чем временная сложность, поскольку процессорное время – это не единственный ресурс, который необходим для выполнения алгоритма. Временная сложность – это количество времени, необходимое на выполнение алгоритма. Другим ресурсом является память, т. е. пространственная сложность – количество памяти, которое требуется для выполнения алгоритма.

Определение: асимптотическая сложность – оценка сложности алгоритма с использованием предельного перехода при стремлении к бесконечности размерности входных данных.

Для обозначения оценки сложности алгоритмов используется O-нотация, которая определяет характеристики функции, показывающей, как изменяется вычислительная сложность алгоритма при изменении количества входных данных в худшем случае.

Введем обозначения: $O(n)$ – оценка количества операций, где n – это количество модулярных оснований на входе алгоритма (m – максимальное основание МСС). В работе количество модулярных оснований равно 5, т. е. $n=5$, максимальное основание МСС $m=32771$.

Поясним на примере операции перевода из модулярной в позиционную систему счисления методом ортогональных базисов, как производится оценка параметров в строках и столбцах табл. 2. Алгоритм метода представлен на рис. 3.

Таблица 2

Оценка сложности методов выполнения немодульных операций в МСС

Перевод из модулярной системы счисления в позиционную систему счисления			
Модулярные основания $p_1=32765,\ p_2=32767, p_3=32768, p_4=32769, p_5=32771$			
Метод ортогональных базисов			
Количество модулярных оснований n	**Операции**	**Оценка количества**	**Общая сложность метода**
5	Умножение	$O(n)$	$O(n^2)$
	Сложение	$O(n)$	

Окончание табл. 2

Количество модулярных оснований n	**Операции**	**Оценка количества**	**Общая сложность метода**
Метод перевода в обобщенную позиционную систему счисления			
5	Вычисление мультипликативной инверсии	$O(n^2)$ сложностью $O\left(\log^2(m)\right)$	$O(n^2 \cdot \log^2(m))$
	Деление по модулю	$O(n^2)$	
	Умножение	$O(n^2)$	
	Сложение	$O(n^2)$	
Сравнение чисел в МСС			
Модулярные основания $p_1 = 32765,\ p_2 = 32767, p_3 = 32768, p_4 = 32769, p_5 = 32771$			
Метод, основанный на переводе чисел из МСС в ПСС			
5	Умножение	$O(n)$	$O(n^2)$
	Сложение	$O(n)$	
Метод с использованием позиционных характеристик			
5	Вычисление мультипликативной инверсии	$O(n)$ сложностью $O\left(\log^2(m)\right)$	$O\left(n \cdot log^2(m)\right)$
	Деление по модулю	$O(n)$	
	Умножение	$O(n)$	
	Сложение	$O(n)$	
	Деление по модулю	$O(n^2)$	
	Умножение	$O(n^2)$	
	Сложение	$O(n^2)$	

Примечание: составлено автором на основании данных, полученных в исследовании.

```
def to_int(self, return_rank=False):
    res = 0
for i, pp in enumerate(self.rns.basis):
    m = self.rns.weight[i]
    b = m * pp
    res += b * self.residues[i]
if return_rank:
```

Рис. 3. Алгоритм перевода из МСС в ПСС методом ортогональных базисов
Примечание: составлено автором на основании данных, полученных в исследовании.

В строке for i, pp in enumerate (self.rns.basis) имеем цикл размера n.

В строке b = m * pp имеем одно умножение.

В строке res += b * self.residues[i] имеем одно умножение и сложение, но поскольку res в общем случае большое число, то сложность этого сложения тоже составляет n. В итоге получаем общую сложность метода $O(n^2)$.

ЗАКЛЮЧЕНИЕ

Рассмотрены алгоритмы перевода чисел из модулярной системы счисления в позиционную систему счисления, а также алгоритмы сравнения чисел в модулярной системе счисления. Алгоритмы используют произвольные наборы модулей, обеспечивающих вычисления в большом динамическом диапазоне. В качестве примера набора оснований МСС были выбраны числа $p_1 = 32765,\ p_2 = 32767,\ p_3 = 32768, p_4 = 32769,\ p_5 = 32771$ с мощностью диапазона:

$$P = 37778931511113441116160.$$

Вычислительный диапазон близок к значению:

$$2^{75} = 37778931862957161709568.$$

Для интерпретации результатов исследования на практике взяты числа

$$A = 2^{32} = 4294967296 = (36, 4, 0, 4, 36),$$
$$B = 2^{30} = 1073741824 = (9, 1, 0, 1, 9).$$

Анализируя операцию перевода из МСС в ПСС и сравнивая методы ортогональных базисов и перевода в ОПСС, отметим, что недостатком первого метода является то, что необходимо выполнять операции с большими числами B_i. Операции сложения и умножения нужно выполнять в ПСС, и если полученный результат выходит за пределы вычислительного диапазона МСС, то необходимо вычислять ранг числа. Преимуществом второго метода является выполнение большинства параллельных вычислений в МСС. Оценивая сложности алгоритмов, получим, что метод ортогональных базисов имеет меньшую сложность.

Рассматривая сравнения чисел в МСС, в частности метод, основанный на переводе чисел из МСС в ПСС, и метод с использованием позиционных характеристик, в первом случае учитывается вычислительная сложность получения остатка от деления на большое число P, во втором – функция *Pirlo* проигрывает КТО, так как требует дополнительных сравнений числа. Первый метод является более эффективным и имеет меньшую асимптотическую вычислительную сложность.

Список источников

1. Лобес М. В. Разработка методов и алгоритмов модулярных вычислений для задач большой алгоритмической сложности : автореф. дис. … канд. физ.-мат. наук. Ставрополь, 2009. 20 с.
2. Лавриненко А. Н., Червяков Н. И. Исследование немодульных операций в àсистеме остаточных классов // Науч. ведомости. Компьютер. моделирование 2012. № 1 (120), Вып. 21/1. С. 110–122.
3. Акушский И. Я., Юдицкий Д. И. Машинная арифметика в остаточных классах. М. : Сов. Радио, 1968. 439 с.
4. Исупов К. С. Методика выполнения базовых немодульных операций в модулярной арифметике с применением интервальных позиционных характеристик // Изв. высш. учеб. заведений. Поволжский регион. Технич. науки. 2013. № 3 (27). С. 26–39.
5. Червяков Н. И., Авербух В. М., Бабенко М. Г. и др. Приближенный метод выполнения немодульных операций в системе остаточных классов // Фундамент. исслед. 2012. № 6–1. С. 189–193. URL: https://fundamental-research.ru/ru/article/view?id=29963 (дата обращения: 20.10.2022).
6. Ляхов П. А., Гробова Т. А., Карасев И. В. Моделирование алгоритмов обратного преобразования чисел из системы остаточных классов в позиционную систему счисления // Актуал. направления науч. исслед. XXI века: теория и практика. 2015. Т. 3, № 7–4 (18–4). С. 430–433.
7. Ляхов П. А., Голошубова Ю. В., Попова Е. А. Сравнительный анализ методов перевода чисел из системы остаточных классов в позиционную систему счисления // Молодой ученый. 2017. № 22 (156). С. 1–6. URL: https://moluch.ru/archive/156/44137/ (дата обращения: 19.10.2022).

References

1. Lobes M. V. Razrabotka metodov i algoritmov moduliarnykh vychislenii dlia zadach bolshoi algoritmicheskoi slozhnosti : Extended abstract of Cand. Sci. Dissertation (Physics and Mathematics). Stavropol, 2009. 20 p. (In Russian).
2. Lavrinenko A. N., Chervyakov N. I. Nomodal Operations Research in System of Residual Classes // Belgorod State University. Scientific Bulletin. Series: Economics. Information Technologies. 2012. No. 1 (120), Is. 21/1. P. 110–122. (In Russian).
3. Akushsky I. Ya., Yuditsky D. I. Mashinnaia arifmetika v ostatochnykh klassakh. Moscow : Sov. Radio, 1968. 439 p. (In Russian).
4. Isupov K. S. Methods of Basic Non-Modular Operations in Modular Arithmetic Using Interval Positional Characteristics // University Proceedings. Volga Region. Technical Sciences. 2013. No. 3 (27). P. 26–39. (In Russian).
5. Chervyakov N. I., Averbukh V. M., Babenko M. G. et al. Approximate Method of Implementation Non-Modular Operations in the Residue Number System // Fundamental Research. 2012. No. 6–1. P. 189–193. URL: https://fundamental-research.ru/ru/article/view?id=29963 (accessed: 20.10.2022). (In Russian).
6. Lyakhov P. A., Grobova T. A., Karasev I. V. Modelirovanie algoritmov obratnogo preobrazovaniia chisel iz sistemy ostatochnykh klassov v pozitsionnuiu sistemu schisleniia // Aktual. napravleniia nauch. Issled. XXI veka: teoriia i praktika. 2015. Vol. 3, No. 7–4 (18–4). P. 430–434. (In Russian).
7. Lyakhov P. A., Goloshubova Yu. V., Popova E. A. Sravnitelnyi analiz metodov perevoda chisel iz sistemy ostatochnykh klassov v pozitsionnuiu sistemu schisleniia // Molodoi uchenyi. 2017. No. 22 (156). P. 1–6. URL: https://moluch.ru/archive/156/44137/ (accessed: 19.10.2022). (In Russian).

8. Полисский Ю. Д. Сравнение чисел в системе остаточных классов // 50 лет модулярной арифметики : тр. Юбилейн. Междунар. науч.-техн. конф., 23–25 ноября 2005 г., Москва, Зеленоград. М. : МИЭТ, 2005. С. 274–290.
9. Краснобаев В. А., Янко А. С., Кошман С. А. Метод арифметического сравнения данных, представленных в системе остаточных классов // Кибернетика и систем. анализ. 2016. Т. 52, № 34. С. 157–162.
10. Бабенко М. Г., Черных А. Н, Червяков Н. И. и др. Эффективное сравнение чисел в системе остаточных классов на основе позиционной характеристики // Тр. ИСП РАН. 2019. Т. 31, Вып. 2. С. 187–202.
11. Исупов К. С. Об одном алгоритме сравнения чисел в системе остаточных классов // Вестн. Астрахан. гос. техн. ун-та. Сер. Управление, вычисл. техн. информ. 2014. № 3. С. 40–49.
12. Тейтельбаум В. Н. Сравнение чисел в чешской системе счисления // Докл. АН СССР. 1958. Т. 121, № 5. С. 807–810.
13. Копыткова Л. Б. Математические модели нейросетевой реализации модулярных вычислительных структур для высокоскоростной цифровой фильтрации : дис. ... канд. физ.-мат. наук. Ставрополь, 2001. 264 с.
14. Червяков Н. И., Ляхов П. А., Копыткова Л. Б. и др. Обработка информации в системе остаточных классов (СОК). Ставрополь : Север.-Кавказ. федер. ун-т, 2016. 225 с. URL: https://book.ru/book/928854 (дата обращения: 06.09.2022).
15. Кочеров Ю. Н. Разработка методов и алгоритмов разделения и восстановления данных в модулярных пороговых структурах для распределенных вычислительных сетей : моногр. Ставрополь : Север.-Кавказ. федер. ун-т, 2016. 239 с.

8. Polissky Yu. D. Sravnenie chisel v sisteme ostatochnykh klassov // 50 let moduliarnoi arifmetiki : Proceedings of the Anniversary International Scientific and Engineering Conference, November 23–25, 2005, Moscow, Zelenograd. Moscow : MIET, 2005. P. 274–290. (In Russian).
9. Krasnobaev V. A., Yanko A. S., Koshman S. A. Metod arifmeticheskogo sravneniia dannykh, predstavlennykh v sisteme ostatochnykh klassov // Kibernetika i system. analiz. 2016. Vol. 52, No. 34. P. 157–162. (In Russian).
10. Babenko M. G., Chernykh A. N., Chervyakov N. I. et al. Efficient Number Comparison in the Residue Number System Based on Positional Characteristics // Proceedings of the Institute for System Programming of the RAS. 2019. Vol. 31, Is. 2. P. 187–202. (In Russian).
11. Isupov K. S. On an Algorithm for Number Comparison in the Residue Number System // Vestnik of Astrakhan State Technical University. Series: Management, Computer Science and Informatics. 2014. No. 3. P. 40–49. (In Russian).
12. Teitelbaum V. N. Comparison of Numbers in the Czechic System of Numbers // Doklady Akademii Nauk SSSR. 1958. Vol. 121, No. 5. P. 807–810. (In Russian).
13. Kopytkova L. B. Matematicheskie modeli neirosetevoi realizatsii moduliarnykh vychislitelnykh struktur dlia vysokoskorostnoi tsifrovoi filtratsii : Cand. Sci. Dissertation (Physics and Mathematics). Stavropol, 2001. 264 p. (In Russian).
14. Chervyakov N. I., Lyakhov P. A., Kopytkova L. B. et al. Obrabotka informatsii v sisteme ostatochnykh klassov (SOK). Stavropol : North-Caucasus Federal University, 2016. 225 p. URL: https://book.ru/book/928854 (accessed: 06.09.2022). (In Russian).
15. Kocherov Yu. N. Razrabotka metodov i algoritmov razdeleniia i vosstanovleniia dannykh v moduliarnykh porogovykh strukturakh dlia raspredeleniia vychislitelnykh setei : Monograph. Stavropol : North-Caucasus Federal University, 2016. 239 p. (In Russian).

Информация об авторе

Н. С. Золотарева – аспирант.

Information about the author

N. S. Zolotareva – Postgraduate.