doi: https://doi.org/10.15407/emodel.42.04.103
УДК 681.04

Ю.Д. Полисский, канд. техн. наук
Научно-исследовательский институт
автоматизации черной металлургии
(Украина, 49000, Днепр, ул. Шевченко, 59,
тел. 0677068311, e-mail: polissky477@gmail.com)

Алгоритм преобразования представлений чисел в системах остаточных классов

Дано теоретическое обоснование одного из подходов к решению немодульной операции, для реализации которой необходимо знание цифр операндов по всем разрядам. Предложенный подход основан на определении остатков по искомым модулям полученных остатков по модулям исходной системы. Такое определение выполняют последовательным вычитанием констант из полученных остатков и суммированием этих констант с результатами, полученными по искомым модулям. При этом константы на каждой итерации выбираются в зависимости от значения остатка в анализируемом разряде. Данный подход целесообразно рассматривать в качестве одного из направлений по исследованию путей повышения эффективности вычислений.

К л ю ч е в ы е с л о в а: остаточные классы, диапазон, модули, преобразование.

Использование непозиционной системы счисления остаточных классов (СОК) позволяет значительно повысить быстродействие операций обработки данных [1]. Система счисления остаточных классов — это система, в которой произвольное число N представлено в виде набора наименьших неотрицательных остатков по модулям $m_1, m_2, \ldots, m_n$, т.е. $N = \left(\alpha_1, \alpha_2, \ldots, \alpha_n\right)$. Здесь $\alpha_i = N \,(\operatorname{mod} m_i)$. При этом, если числа m_i взаимно простые, то такому представлению соответствует только одно число N в диапазоне $\left[0, M\right)$, где $M = m_1 m_2 \ldots m_n$.

При обработке данных, представленных в системе СОК, существенное место занимают промежуточные преобразования чисел из одной системы остаточных классов в другую [2]. Решение такой задачи может потребоваться, например, при расширении диапазона представления чисел, определении ранга числа, модульном делении чисел в тех случаях, когда осуществляется деление на число, кратное одному или нескольким модулям системы. Поэтому операция перехода от одной системы модулей к другой, при которой по известным остаткам числа в одной системе СОК определяют значения остатков этого же числа в другой, относится к одной из основных немодульных операций в системе СОК.

Впервые подход к решению данной задачи предложен в классической работе [1]. Результаты дальнейших исследований [2] свидетельствуют о возможности получения более экономных решений. Однако алгоритм, предложенный в работе [2], имеет значительное число разнотипных шагов, что увеличивает время его реализации.

Предлагаемое новое алгоритмическое решение позволяет существенно упростить практическую реализацию и ускорить получение результата. Предлагается также теоретическое обоснование данного подхода к повышению эффективности выполнения в непозиционной системе СОК немодульной, так называемой сложной, операции, для реализации которой необходимо знание цифр операндов по всем разрядам. Операция заключается в преобразовании представления числа одной системой модулей представлением его другой системой модулей.

Постановка задачи. Пусть имеются две системы СОК,

$$C1 = \{m_1, m_2, \ldots, m_n\},\ M_1 = m_1 m_2 \ldots m_n, \tag{1}$$

$$C2 = \{p_1, p_2, \ldots, p_r\},\ M_2 = p_1 p_2 \ldots p_r, \tag{2}$$

и число

$$N = (\alpha_1, \alpha_2, \ldots, \alpha_n) \tag{3}$$

такое, что $N = (\alpha_1, \alpha_2, \ldots, \alpha_n) < \min\{M_1, M_2\}$. Необходимо для представления числа (3) в системе СОК (1) выполнить преобразование, т.е. представление $N = (\beta_1, \beta_2, \ldots, \beta_r)$ этого числа в системе СОК (2), где $\beta_i = N \pmod{p_i}$.

Рассмотрим алгоритм построения и его табличную реализацию. Если системой оснований полиадического кода является система $m_1, m_2, \ldots, m_n$, то число N в полиадическом коде имеет вид

$$N = \pi_1 + \pi_2 m_1 + \pi_3 m_1 m_2 + \ldots + \pi_i m_1 m_2 \ldots m_{i-1} + \cdots$$
$$\ldots + \pi_{n-1} m_1 m_2 \ldots m_{n-2} + \pi_n m_1 m_2 \ldots m_{n-1},$$

где $0 < \pi_i \le m_i - 1$. Отсюда

$$\beta_i = N (\mathrm{mod}\, p_i) =$$
$$= (\pi_1 + \pi_2 m_1 + \ldots + \pi_i m_1 m_2 \ldots m_{i-1} + \cdots + \pi_n m_1 m_2 \ldots m_{n-1}) (\mathrm{mod}\, p_i).$$

Тогда

$$\alpha_i = (\pi_1 + \pi_2 m_1 + \cdots + \pi_{n-1} m_1 m_2 \ldots m_{n-2} + \pi_n m_1 m_2 \ldots m_{n-1}) (\mathrm{mod}\, m_i),$$

$$\alpha_1 = N\left(\bmod m_1\right) = \pi_1 \,, \pi_1 = \alpha_1 \,,$$

$$\alpha_2 = N\left(\bmod m_2\right) = \pi_1 + \pi_2 m_1 \cdot \, \pi_2 m_1 = \alpha_2 - \pi_1 \,,$$

. . . .

. . . .

$$\alpha_i = (\pi_1 + \pi_2 m_1 + \pi_3 m_1 m_2 + \ldots + \pi_{i-1} m_1 m_2 \ldots m_{i-2} + \pi_i m_1 m_2 \ldots m_{i-1})(\bmod m_i)\,,$$

$$\pi_i m_1 m_2 \ldots m_{i-1} = \alpha_i - (\pi_1 + \pi_2 m_1 + \pi_3 m_1 m_2 + \ldots + \pi_{i-1} m_1 m_2 \ldots m_{i-2})(\bmod m_i)\,.$$

Отсюда следует, что определение очередного слагаемого возможно только после получения предыдущего.

Таким образом, алгоритм имеет следующие итерации.

Принимаем значения искомых остатков $\beta_1 = 0,\ \beta_2 = 0, \ldots, \beta_r = 0$. На первой итерации по каждому из модулей $m_1, m_2, \ldots, m_{n-1}$ определяем разность между остатком по этому модулю и значением π_1,, взятую по данному модулю. В результате получаем $\tilde{N} = \left(\tilde{\alpha}_1 = 0, \tilde{\alpha}_2, \ldots, \tilde{\alpha}_n\right)$ и значение $\pi_2 m_1$. Назовем такие разности приведенными остатками. В то же время, значение π_1 суммируем с $\beta_j,\ j = 1, 2, \ldots, r$.

На второй итерации по каждому из модулей $m_1, m_2, \ldots, m_{n-1}$ определяем разность между приведенным остатком по этому модулю и значением $\pi_2 m_1$, взятую по данному модулю. В результате находим $\tilde{N} = \left(\tilde{\alpha}_1 = 0, \tilde{\alpha}_2 = 0, \tilde{\alpha}_3, \ldots, \tilde{\alpha}_n\right)$ и значение $\pi_3 m_1 m_2$. Значение $\pi_2 m_1$ суммируем с полученными ранее значениями остатков $\beta_j,\ j = 1, 2, \ldots, r$. После выполнения n итераций получаем $\tilde{\alpha}_1 = 0, \tilde{\alpha}_2 = 0, \ldots, \tilde{\alpha}_n = 0$ и значения искомых остатков $\beta_1, \beta_2, \ldots, \beta_r$.

Табличная реализация алгоритма заключается в задании на данной итерации значений позиционной характеристики π_i, $\pi_i = 0, 1, \ldots, m_i - 1$, и определении на их основе приведенных остатков $\tilde{\alpha}_i$ и констант Δ^i в соответствии с зависимостями

$$\Delta^i = \pi_i m_1 m_2 \ldots m_{i-1},\ \ \tilde{\alpha}_i = (\Delta^i)(\bmod m_i),\ \ \Delta^i_i = (\Delta^i)(\bmod m_i),$$

$$\Delta^i_{i+1} = (\Delta^i)(\bmod m_{i+1}),\ \ \Delta_n = (\Delta^i)(\bmod m_n).$$

Для каждого значения приведенного остатка константы получаются с помощью простой их выборки из соответствующей таблицы.

Рассмотрим переход от представления числа $N^1{}_1 = 97 = (2,6,1)$ в системе модулей $m_1 = 5,\ m_2 = 7,\ m_3 = 3$, $M_1 = m_1 m_2 m_3 = 5*7*3 = 105$ к его представлению в системе модулей $p_1 = 11,\ p_2 = 13$, $M_2 = p_1 p_2 = 11*13 = 143$, $M_2 > M_1$. До определения значений остатков β_1 и β_2 примем их начальные значения равными нулю, т.е. $N^1{}_2 = (0,0)$.

На первой итерации из табл.1 для $\alpha_1^1 = 2$ выбираем константы 2, 2, 2 соответственно по модулям $m_1 = 5,\ m_2 = 7,\ m_3 = 3$, которые вычитаем из остатков 2, 6, 1. Получаем $N_1^1 = 95 = (0,4,2)$. Для $\alpha_1^1 = 2$ выбираем константы 2, 2 соответственно по модулям $p_1 = 11,\ p_2 = 13$, которые прибавляем к остаткам 0, 0. В результате получаем $N_2^1 = (2,2)$.

На второй итерации из табл. 2 для $\alpha_2^2 = 4$ выбираем константы 4, 1 соответственно по модулям $m_2 = 7, m_3 = 3$, которые вычитаем из остатков 4, 2. Получаем $N_1^2 = 70 = (0,0,1)$. Для $\alpha_2^2 = 5$ выбираем константы 5, 5 соответственно по модулям $p_1 = 11,\ p_2 = 13$, которые прибавляем к остаткам 5, 2. В результате получаем $N_2^2 = (5,1)$.

На третьей итерации из табл. 3 для $\alpha_3^3 = 1$ выбираем константу 1 по модулю $m_3 = 3$, которую вычитаем из остатка 1. Получаем $N_1^3 = 0 = (0,0,0)$. Для $\alpha_3^3 = 1$ выбираем константы 4, 5 соответственно по модулям $p_1 = 11,\ p_2 = 13$, которые прибавляем к остаткам 5,1. В результате получаем $N_2^3 = 97 = (9,6)$.

Таблица 1

π_1	Δ_1	α_1^1	α_2^1	α_3^1	β_1	β_2
По модулям						
5			7	3	11	13
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	2	2	2	2	2	2
3	3	3	3	0	3	3
4	4	4	4	1	4	4

Таблица 2

π_2	$\Delta_2 = \pi_2 m_1$	α_2^2	α_2^2	β_1	β_2
По модулям					
7			3	11	13
0	0	0	0	0	0
1	5	5	2	5	5
2	10	3	1	10	10
3	15	1	0	4	2
4	20	6	2	9	7
5	25	4	1	3	12
6	30	2	0	8	4

Для увеличения быстродействия алгоритмов было введено представление чисел в обратном коде [3]. При задании числа N остатками число $\overline{N} = \left(\overline{\alpha}_1, \overline{\alpha}_2, \ldots, \overline{\alpha}_n\right)$ является представлением числа $N = \left(\alpha_1, \alpha_2, \ldots, \alpha_n\right)$ в обратном коде, где $\overline{\alpha}_i = (m_i - 1) - \alpha_i$ — обратный код остатка α_i.

Обратным кодом числа $N_1^1 = 97$ является число $\overline{N}_1^1 = 7$. Рассмотрим изложенное на примере перехода от представления числа $\overline{N}_1^1 = 7 = (2,0,1)$ в системе модулей $m_1 = 5,\ m_2 = 7,\ m_3 = 3$, $M_1 = m_1 m_2 m_3 = 5 * 7 * 3 = 105$ к его представлению в системе модулей $p_1 = 11,\ p_2 = 13$, $M_2 = p_1 p_2 = = 11 * 13 = 143$, $M_2 > M_1$. До определения значений остатков $\overline{\beta}_1$ и $\overline{\beta}_2$ примем их начальные значения равными нулю, т.е. $\overline{N}_2^1 = (0,0)$.

На первой итерации из табл. 1 для $\overline{\alpha}_1^1 = 2$ выбираем константы 2, 2, 2 соответственно по модулям $m_1 = 5,\ m_2 = 7,\ m_3 = 3$, которые вычитаем из остатков 2, 0, 1. Получаем $\overline{N}_1^1 = 5 = (0,5,2)$. Из этой же табл. 1 для $\overline{\alpha}_1^1 = 2$ выбираем константы 2, 2 соответственно по модулям $p_1 = 11,\ p_2 = 13$, которые прибавляем к остаткам 0, 0. В результате получаем $\overline{N}_2^1 = (2,2)$.

На второй итерации из табл. 2 для $\overline{\alpha}_2^2 = 5$ выбираем константы 5, 2 соответственно по модулям $m_2 = 7,\ m_3 = 3$, которые вычитаем из остатков 5, 2. Получаем $\overline{N}_1^2 = 70 = (0,0,0)$. Из этой же табл. 2 для $\overline{\alpha}_2^2 = 5$ выбираем константы 5, 5 соответственно по модулям $p_1 = 11,\ p_2 = 13$, которые прибавляем к остаткам 2, 2. В результате получаем $\overline{N}_2^2 = (7,7)$.

Для перехода представления числа $N_1^1 = 97 = (2,6,1)$ в системе модулей $m_1 = 5, m_2 = 7, m_3 = 3$, $M = mmm = 5 * 7 * 3 = 105$ к его представлению в системе модулей $p_1 = 11,\ p_2 = 13$, $M_2 = p_1 p_2 = 11 * 13 = 143$, $M_2 > M_1$ выполняем заключительную операцию $N_2^3 = (M_1 - 1) - \overline{N}_2^2 = (5,0) - (7,7) = = (9,6)$. Поскольку при представлении числа в прямом и обратном кодах для достижения результата требуется различное число итераций, целесообразно выполнять преобразование одновременно для двух этих представлений.

Таблица 3

π_3	$\Delta_3 = \pi_3 m_1 m_2$	α_3^3	β_1	β_2
По модулям				
3			11	13
0	0	0	0	0
1	35	2	2	9
2	70	1	4	5

Искомый результат определяется по значениям $\tilde{\alpha}_i$, $i = 1, 2, \ldots, n$, того из представлений, для которого первым получен результат ($\tilde{\alpha}_1 = 0,$ $\tilde{\alpha}_2 = 0, \ldots, \tilde{\alpha}_n = 0$). При этом верхняя оценка числа итераций алгоритма равна числу модулей исходной системы, что значительно меньше временной оценки алгоритма [2].

Таким образом, рассмотренный алгоритм обеспечивает преобразование представлений чисел в системах остаточных классов.

Выводы

Обработка данных в непозиционной системе счисления остаточных классов позволяет существенно повысить быстродействие вычислительных операций. Предложенный подход к решению немодульной операции перехода от представления числа одной системой модулей к его представлению другой системой модулей основан на определении остатков по искомым модулям исходной системы. Данный подход можно рассматривать как одно из направлений исследования способов повышения эффективности вычислений.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. *Акушский И.Я., Юдицкий Д.И.* Машинная арифметика в остаточных классах. М.: Сов. радио, 1968, 440 с.
2. *Магомедов Ш.Г.* Преобразование представлений чисел в модулярной арифметике в системах остаточных классов с разными основаниями//Вестник АГТУ. Сер.: Управление, вычислительная техника и информатика, 2014, № 4, с. 32—39.
3. *Полисский Ю.Д.* Алгоритм выполнения сложных операций в системе остаточных классов с помощью представления чисел в обратных кодах// Электрон. моделирование, 2014, **36**, № 4, с. 117–123

Получена 12.06.20;
после доработки 24.06.20

REFERENCES

1. Akushskiy, I.Ya. and Yudickiy, D.I. (1968), *Mashinnaya arifmetika v ostatochnykh klassakh* [Machine arithmetic in the residual classes], Sov. Radio, Moscow, USSR.
2. Magomedov, Sh.G. (2014), “Transformation of representations of numbers in modular arithmetic in systems of residual classes with different bases”, *Vestnik AGTU. Ser.: Upravleniye, vychislitel'naya tekhnika i informatika*, no. 4, pp. 32-39.
3. Polissky, Yu.D. (2014), “An algorithm for performing complex operations in the residual class system using the representation of numbers in reverse codes”, *Elektron. modelirovaniye*, Vol. 36, no. 4, pp. 117-122.

Received 12.06.20;
afterrevision 24.06.20

Ю.Д. Поліський

АЛГОРИТМ ПЕРЕТВОРЮВАННЯ ПРЕДСТАВЛЕНЬ
ЧИСЕЛ У СИСТЕМАХ ЗАЛИШКОВИХ КЛАСІВ

Дано теоретичне обґрунтування одного з підходів до вирішення немодульної операції, для реалізації якої необхідне знання цифр операндів за всіма розрядами. Запропонований підхід основано на визначенні залишку стосовно шуканих модулів отриманих залишків по модулях вихідної системи. Таке визначення виконують послідовним відніманням констант від отриманих залишків і подсумовуванням їх до результатів, які утворюються стосовно шуканих модулів. При цьому константи на кожній ітерації вибираються в залежності від значення залишку в аналізованому розряді. Даний підхід доцільно розглядати як один з напрямків дослідження шляхів підвищення ефективності обчислень.

К л ю ч о в і с л о в а: залишкові класи, діапазон, модулі, перетворення.

Yu.D. Polissky

ALGORITHM FOR TRANSFORMING REPRESENTATIONS
OF NUMBERS IN SYSTEMS OF RESIDUAL CLASSES

Data processing in a non-positional number system of residual classes can significantly improve the speed of computing operations. The aim of the work is the theoretical justification of one of the approaches to solving a non-modular, so-called complex, operation, for the implementation of which it is necessary to know the numbers of operands for all digits. The operation consists in transforming the representation of a number from one system of modules by representing it in another system of modules. The considered approach is based on determining the remainder by the desired module based on the obtained balances by the modules of the original system. Such a determination is performed by sequentially subtracting the constants from the obtained residues and summing these constants to the results that are generated by the desired module. In this case, the constants at each iteration are selected depending on the value of the remainder in the analyzed discharge. It is advisable to consider this approach as one of the directions for studying ways to increase the efficiency of calculations.

K e y w o r d s: residual classes, range, modules, conversion.

ПОЛИССКИЙ Юрий Давидович, канд. техн. наук, ст. науч. сотр. Научно-исследовательского института автоматизации черной металлургии, г. Днепр. В 1960 г. окончил Днепропетровский металлургический институт. Область научных исследований — системы и средства управления.