DOI 10.34185/1562-9945-3-128-2020-14
УДК 681.04

Ю.Д. Полисский

ОБ ОДНОМ АЛГОРИТМИЧЕСКОМ РЕШЕНИИ ЗАДАЧИ ВОССТАНОВЛЕНИЯ ОСТАТКА ЧИСЛА В СИСТЕМЕ ОСТАТОЧНЫХ КЛАССОВ

Аннотация. При выполнении операций расширения диапазона представления чисел, деления, определения переполнения, масштабирования, контроля ошибок вычислений возникает задача восстановления остатка числа по данному модулю на основании остатков этого числа по остальным модулям системы. Табличное выполнение операции восстановления остатка числа реализуется с помощью базового алгоритма. Метод решения основан на определении остатка по данному модулю на основании полученных остатков по остальным модулям системы. Такое определение выполняется последовательным вычитанием констант из полученных остатков и суммированием этих констант к результататам, которые формируются по данному модулю. При этом константы на каждой итерации выбираются в зависимости от значения остатка в анализируемом разряде. При несомненном достоинстве метода сохраняются требования к быстродействию выполнения операции восстановления остатка числа. Целью исследования является аналитическое рассмотрение подхода к ускоренной реализации базовой операции восстановления остатка числа по данному модулю на основании остатков этого числа по остальным модулям системы. Одна из реализаций алгоритма состоит в одновременном его выполнении по базовому варианту для искомого числа и числа, обратного искомому. При этом искомый остаток определяется по значению остатка того из чисел, для которого первым получается результат поиска. Приведены варианты реализации алгоритма с переходами от представления числа в прямом коде к представлению этого числа в обратном коде и от представления числа в обратном коде к его представлению в прямом коде. Рассмотренный алгоритм реализации в системе остаточних классов базовой немодульной операции восстановления значения остатка числа по данному модулю на основании значений остатков этого числа по остальным модулям системы обеспечивает получение искомого результата. На основе предложенных подходов достигается ускоренная реализация базовой операции восстановления остатка числа по данному модулю. Представляется целесообразным применить предложенные подходы в качестве перспективных направлений исследований этой операции в системе остаточных классов.

Ключевые слова: система остаточных классов, сложные операции, полиадический код, алгоритм.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Постановка проблемы. Одно из перспективных направлений повышения производительности вычислительных структур связано с использованием непозиционной системы счисления остаточных классов (СОК) [1]. СОК называется система счисления, в которой произвольное число N представляется в виде набора наименьших неотрицательных остатков по модулям $m_1, m_2, \ldots, m_n$, т.е. $N = (\alpha_1, \alpha_2, \ldots, \alpha_n)$. Здесь $\alpha_i = N(\operatorname{mod} m_i)$. При этом, если числа m_i взаимно простые, то такому представлению соответствует только одно число N диапазона $[0, M)$, где $M = m_1 m_2 \ldots m_n$.

Пусть системой оснований полиадического кода также является система $m_1, m_2, \ldots, m_n$. Тогда число N в полиадическом коде представляется следующим образом

$N = \pi_1 + \pi_2 m_1 + \pi_2 m_1 m_2 + \ldots + \pi_i m_1 m_2 \ldots m_{i-1} + \ldots + \pi_{n-1} m_1 m_2 \ldots m_{n-2} + \pi_n m_1 m_2 \ldots m_{n-1}$, где π_i - i -я позиционная характеристика $0 \le \pi_i < m_i - 1, i = 1, 2, \ldots, n$.

Существенным недостатком СОК является сложность выполнения немодульных операций. В частности, такой операции, как восстановление остатка числа по данному модулю на основании остатков этого числа по остальным модулям системы. Данная операция необходима для выполнения операций расширения диапазона представления чисел, деления, определения переполнения, масштабирования, контроля ошибок вычислений.

Анализ последних исследований и публикаций. Табличное выполнение операции восстановления остатка числа реализуется с помощью базового алгоритма [2-4]. Метод решения основан на определении остатка по данному модулю на основании полученных остатков по остальным модулям системы. Такое определение выполняется последовательным вычитанием констант из полученных остатков и суммированием этих констант к результатататам, которые формируются по данному модулю. При этом константы на каждой итерации выбираются в зависимости от значения остатка в анализируемом разряде. При несомненном достоинстве метода сохраняются требования к быстродействию выполнения операции восстановления остатка числа.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Цель исследования. Целью исследования является аналитическое рассмотрение подхода к ускоренной реализации базовой операции восстановления остатка числа по данному модулю на основании остатков этого числа по остальным модулям системы.

Изложениее основного материала. Базовый алгоритм состоит из следующих итераций. Значение восстанавливаемого остатка принимается $\alpha_n = 0$. На первой итерации по каждому из модулей $m_1 m_2 \ldots m_{n-1}$ определяется разность между остатком по этому модулю и значением π_1 , взятая по данному модулю, в результате чего находится значение $\pi_2 m_1$. Назовем такие разности приведенными остатками. В то же время значение π_1 суммируется с α_n. На второй итерации по каждому из модулей $m_2 \ldots m_{n-1}$ определяется разность между приведенным остатком по этому модулю и значением $\pi_2 m_1$, взятая по данному модулю. В результате находится значение $\pi_3 m_1 m_2$, а значение $\pi_2 m_1$ суммируется с полученным раннее значением остатка по модулю m_1. После выполнения $(n-1)$ итераций получаем $(\ddot{\alpha}_1 = 0, \tilde{\alpha}_2 = 0, \ldots, \tilde{\alpha}_n)$. Табличная реализация алгоритма заключается в задании на данной итерации значений позиционной характеристики π_i, $\pi_i = 0, 1, 2, \ldots, m_i - 1$ и определении на их основе приведенных остатков $\tilde{\alpha}_i$ и констант Δ^i в соответствии с зависимостями $\Delta^i = \pi_i m_1 m_2 \ldots m_{i-1}$, $\tilde{\alpha}_i = (\Delta^i)(\mathrm{mod}\, m_i)$, $\Delta_i^i = (\Delta^i)(\mathrm{mod}\, m_i)$, $\Delta_{i+1}^i = (\Delta^i)(\mathrm{mod}\, m_{i+1})$, $\Delta_n = (\Delta^i)(\mathrm{mod}\, m_n)$. Для каждого значения приведенного остатка константы получаются путем простой их выборки из соответствующей таблицы.

Для увеличения быстродействия ряда алгоритмов автором было введено представление чисел одновременно в прямом и обратном кодах с возможностью перехода в процессе работы алгоритма от одного представления к другому [5].

При задании числа N остатками число $\bar{N} = (\bar{\alpha}_1, \bar{\alpha}_2, \ldots, \bar{\alpha}_n)$ является представлением числа $N = (\alpha_1, \alpha_2, \ldots, \alpha_n)$ в обратном коде, где $\bar{\alpha}_i = (m_i - 1) - \alpha_i$ - обратный код остатка α_i. При задании числа N в полиадическом коде число

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

$$\bar{N} = \bar{\pi}_1 + \bar{\pi}_2 m_1 + \ldots + \bar{\pi}_i m_1 m_2 \ldots m_{i-1} + \cdots + \bar{\pi}_{n-1} m_1 m_2 \ldots m_{n-2} + \bar{\pi}_n m_1 m_2 \ldots m_{n-1}$$

является представлением числа

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

в обратном коде, где $\bar{\pi}_i = (m_i - 1) - \pi_i$ - обратный код позиционной характеристики π_i. Для рассмотрения работы алгоритма, например, в системе модулей $m_1 = 5, m_2 = 7, m_3 = 3, m_4 = 2, m_5 = 13$ необходимы представленные ниже таблицы констант табл.1, табл.2, табл.3, табл.4.

Таблица 1

Модули						
5			**7**	**3**	**2**	**13**
π_1	$\tilde{\alpha}_1$	Δ_5^1	Δ_7^1	Δ_3^1	Δ_2^1	Δ_{13}^1
0	0	0	0	0	0	0
		4	4	1	0	4
1	1	1	1	1	1	1
		3	3	0	1	3
2	2	2	2	2	0	2
		2	2	2	0	2
3	3	3	3	0	1	3
		1	1	1	1	1
4	4	4	4	1	0	4
		0	0	0	0	0

Таблица 2

Модули					
7			**3**	**2**	**13**
π_2	$\tilde{\alpha}_2 = \pi_2 m_1$	Δ_7^2	Δ_3^2	Δ_2^2	Δ_{13}^2
0	0	0	0	0	0
		6	2	0	7
1	5	5	2	1	5
		1	0	1	2
2	3	3	1	0	10
		3	1	0	10
3	1	1	0	1	2
		5	2	1	5
4	6	6	2	0	7
		0	0	0	0
5	4	4	1	1	12
		2	0	0	4
6	2	2	0	0	4
		4	1	1	12

Таблица 3

Модули				
3			**2**	**13**
π_3	$\tilde{\alpha}_3 = \pi_3 m_1 m_2$	Δ_3^3	Δ_2^3	Δ_{13}^3
0	0	0	0	0
		2	1	9
1	2	2	1	9
		0	0	0
2	1	1	0	5
		1	0	5

Таблица 4

Модули			
2			**13**
π_4	$\tilde{\alpha}_4 = \pi_4 m_1 m_2 m_3$	Δ_2^4	Δ_{13}^4
0	0	0	0
		1	1
1	1	1	1
		0	0

Таблица 5

Модули					
Число	**5**	**7**	**3**	**2**	**13**
	Остатки				
168	3	0	0	0	***0***
—	3	3	0	1	3
	0	4	0	1	3
—	=	4	1	1	12
	=	0	2	0	2
—	=	=	2	1	9
	=	=	0	1	11
—	=	=	=	1	1
	=	=	=	0	**12**

Таблица 6

Модули					
Число	**5**	**7**	**3**	**2**	**13**
	Остатки				
41	1	6	2	1	***0***
—	1	1	1	1	1
	0	5	1	0	1
—	=	5	2	1	5
	=	0	2	1	6
—	=	=	2	1	9
	=	=	0	0	**2**

В этих таблицах для каждого из модулей, представленных своими остатками по модулю соответствующей таблицы, верхняя строка – величина остатка в прямом коде, нижняя строка – в обратном коде.

Одной из реализаций алгоритма может быть одновременное его выполнение по базовому варианту, например, для числа $N = 168$ и обратного ему числа $\bar{N} = (M-1) - N = 209 - 168 = 41$ согласно табл.5 и табл.6 для той же системы модулей $m_1 = 5, m_2 = 7, m_3 = 3, m_4 = 2, m_5 = 13$. При этом искомый остаток определяется по значению $\tilde{\alpha}_n$ того из чисел, для которого первым получен результат $(\tilde{\alpha}_1 = 0, \tilde{\alpha}_2 = 0, \ldots, \tilde{\alpha}_n)$.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

В рассмотренном примере вначале получено $\left(\bar{\tilde{\alpha}}_1 = 0, \bar{\tilde{\alpha}}_2 = 0, \ldots, \bar{\tilde{\alpha}}_n = 2\right)$.

Переход от остатка $\bar{\tilde{\alpha}}_n = \bar{\tilde{\alpha}}_{13}$ к остатку $\tilde{\alpha}_n = \tilde{\alpha}_{13}$ выполняется следующим образом. Пусть $\tilde{\alpha}_n^N = N\left(\operatorname{mod} m_n\right)$ и $\bar{\tilde{\alpha}}_n^N = ((M-1)-N)\left(\operatorname{mod} m_n\right)$.

Тогда $\tilde{\alpha}_n^{M-1} = (M-1)\left(\operatorname{mod} m_n\right)$ и $\tilde{\alpha}_n^{M-1} = (\tilde{\alpha}_n^N + \bar{\tilde{\alpha}}_n^N)(\operatorname{mod} m_n)$.

Остаток $\tilde{\alpha}_n^{M-1} = (M-1)\left(\operatorname{mod} m_n\right) = 209(\operatorname{mod} 13) = 1$. Остаток $\bar{\tilde{\alpha}}_{n+1}^N = 2$.

Следовательно, $\tilde{\alpha}_{n+1}^N = (1-2)(\operatorname{mod} 13) = 12$.

Табл.7, табл.8, табл.9 иллюстрируют варианты реализации алгоритма с переходами от представления числа в прямом коде (Блок А) к представлению этого числа в обратном коде (Блок Б) и от представления числа в обратном коде к его представлению в прямом коде.

При этом для Блока А $\begin{array}{l}\tilde{\alpha}_{i,t} = (\tilde{\alpha}_{i,t-1} - \Delta^{i,t})(\operatorname{mod} m_i), \\ i = 1,2,\ldots,n-1, t = 1,2,\ldots,t-1\end{array}$, $\begin{array}{l}\tilde{\alpha}_{n,t} = (\tilde{\alpha}_{n,t-1} + \Delta^{n,t})(\operatorname{mod} m_i), \\ t = 1,2,\ldots,t-1\end{array}$.

Для блока Б $\begin{array}{l}\tilde{\alpha}_{i,t} = (\tilde{\alpha}_{i,t-1} + \Delta^{i,t})(\operatorname{mod} m_i), \\ i = 1,2,\ldots,n-1, t = 1,2,\ldots,t-1\end{array}$, $\begin{array}{l}\tilde{\alpha}_{n,t} = (\tilde{\alpha}_{n,t-1} - \Delta^{n,t})(\operatorname{mod} m_i) \\ t = 1,2,\ldots,t-1\end{array}$.

Здесь t – количество итераций. В рассмотренном примере целесообразней реализация алгоритма при одновременном его выполнении по базовому варианту для прямого и обратного чисел.

Таблица 7

Блок А							Блок Б					
Модули							Модули					
Число	**5**	**7**	**3**	**2**	**13**		Число	**5**	**7**	**3**	**2**	**13**
	Остатки							Остатки				
168	3	0	0	0	*0*		**41**	1	6	2	1	*0*
—	3	3	0	1	3							
	0	4	0	1	3	→		=	2	2	0	9
							—	=	4	1	1	12
	=	=	2	0	2	←		=	6	0	1	10
—			2	1	9							
			0	1	11							
—				1	1							
				0	**12**							

Таблица 8

Блок А							Блок Б					
Модули							Модули					
Число	**5**	**7**	**3**	**2**	**13**		Число	**5**	**7**	**3**	**2**	**13**
	Остатки							Остатки				
168	3	0	0	0	***0***		**41**	1	6	2	1	***0***
—	3	3	0	1	3							
	0	4	0	1	3	→		=	2	2	0	9
							—	=	4	1	1	12
								=	6	0	1	10
							—	=	=	2	1	9
			0	1	11	←			=	2	0	1
—				1	1							
					12							

Таблица 9

Блок А							Блок Б					
Модули							Модули					
Число	**5**	**7**	**3**	**2**	**13**		Число	**5**	**7**	**3**	**2**	**13**
	Остатки							Остатки				
168	3	0	0	0	***0***	→	**41**	1	6	2	1	***12***
							—	3	3	0	1	3
	=	4	0	1	3	←		4	2	2	0	9
—		4	1	1	12							
		0	2	0	2	→		=	=	0	1	10
							—			2	1	9
				1	11	←				2	0	1
—				1	1							
					12							

В рассмотренном табл.10 и табл.11 примере вначале получено $(\tilde{\alpha}_1 = 0, \tilde{\alpha}_2 = 0, \ldots, \tilde{\alpha}_n = 8)$.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Таблица 10

Модули					
Число	**5**	**7**	**3**	**2**	**13**
	Остатки				
8	3	1	2	0	***0***
–	3	3	0	1	3
	0	5	2	1	3
–	=	5	2	1	5
	=	0	0	0	**8**

Таблица 11

Модули					
Число	**5**	**7**	**3**	**2**	**13**
	Остатки				
201	1	5	0	1	***0***
–	1	1	1	1	1
	0	4	2	0	1
–	=	4	1	1	12
	=	0	1	1	0
–	=	=	1	0	5
	=	=	0	1	5
–	=	=	=	1	1
	=	=	=	0	**6**

Табл.12, табл.13 иллюстрируют варианты реализации алгоритма с переходами от представления числа в прямом коде (Блок А) к представлению этого числа в обратном коде (Блок Б) и от представления числа в обратном коде к его представлению в прямом коде.

Таблица 12

Модули							Модули					
Число	**5**	**7**	**3**	**2**	**13**		Число	**5**	**7**	**3**	**2**	**13**
	Остатки							Остатки				
8	3	1	2	0	***0***		**201**	1	5	0	1	***0***
–	3	3	0	1	3							
	0	5	2	1	3	→		=	1	0	0	9
						←	–	=	5	2	1	5
	=	=	0	0	**8**			=	6	2	1	4

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Таблица 13

Блок А							Блок Б					
Модули							Модули					
Число	**5**	**7**	**3**	**2**	**13**		Число	**5**	**7**	**3**	**2**	**13**
	Остатки							Остатки				
8	3	1	2	0	***0***	→	**201**	1	5	0	1	***12***
						←	–	3	3	0	1	3
	=	5	2	1	3			4	1	0	0	9
–		5	2	1	5							
		0	0	0	**8**							

В рассмотренном примере целесообразна реализация алгоритма как при одновременном его выполнении по базовому варианту для прямого и обратного чисел, так и с переходами от представления числа в прямом коде к представлению этого числа в обратном коде и от представления числа в обратном коде к его представлению в прямом коде.

Выводы. Рассмотрен алгоритм реализации в системе остаточных классов базовой немодульной операции восстановления значения остатка числа по данному модулю на основании значений остатков этого числа по остальным модулям системы. Показано, что проанализированный метод обеспечивает получение искомого результата. На основе предложенных подходов достигается ускоренная реализация базовой операции восстановления остатка числа по данному модулю. Представляется целесообразным применить предложенные подходы в качестве перспективных направлений исследований этой операции в системе остаточных классов.

ЛІИТЕРАТУРА / ЛІТЕРАТУРА

1. Акушский И.Я., Юдицкий Д.И. Машинная арифметика в остаточных классах. - М.: Советское радио, 1968. 440 с.

2. Поліський Ю.Д. Про один метод розширення діапазону зображення чисел у системі залишкових класів. Математичне моделювання. Кам'янське. 2007. №2(17). С. 16–17.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

3. Полисский Ю.Д. Табличная реализация базового алгоритма для выполнения сложных операций в системе остаточных классов. Проблеми математичного моделювання: матеріали наук.-метод.конф. 19-21 трав.2010 р. Кам'янське. 2010. С. 196–199.
4. Полисский Ю.Д. Об ускоренном выполнении сложных операций в модулярных вычислительных структурах. Моделирование-2010, SIMULATION-2010: материалы межнар.науч.конф. 21-23 мая 2010г. Киев 2010. Т.3. С.32-39.
5. Полисский Ю.Д. Алгоритм выполнения сложных операций в системе остаточных классов с помощью представления чисел в обратных кодах. Электронное моделирование. Киев. 2014. №4 Т. 36. С. 117–123

REFERENCES

1. Akushskiy I.Ya., Yudickiy D.I. Machine arithmetic in the residual classes. - M.: Sov. Radio, 1968. 440 p.
2. Polissky Yu.D. About one method of extending the range of the image of numbers in a system of residual classes Mathematical modeling. Kamianske. 2007. №2 (17). pp. 16–17.
3. Polissky Yu.D. A table implementation of the basic algorithm for performing complex operations in a system of residual classes. Problems of mathematical modeling: materials of sciences.-method.konf. 19-21 May. 2010. Kam'yanske. 2010. pp. 196–199.
4. Polissky Yu.D. On the accelerated implementation of complex operations in modular computing structures. Modeling-2010, SIMULATION-2010: materials of the international scientific conference. May 21-23, 2010. Kiev 2010.V.3. pp.2-39.
5. Polissky Yu.D. An algorithm for performing complex operations in a system of residual classes using the representation of numbers in reverse codes. Electronic modeling. Kiev. 2014. No4 V.36. pp .117–123.

Received 02.03.2020.
Accepted 04.03.2020.

Об одном алгоритмическом решении задачи восстановления остатка числа в системе остаточных классов

Рассматривается определение остатка по данному модулю на основании остатков по остальным модулям системы. Определение выполняется последовательным вычитанием (добавлением) констант из полученных остатков и суммированием (вычитанием) этих констант к результататам, которые формируются по данному модулю. Константы на каждой итерации выбираются в зависимости от значения остатка в анализируемом разряде. Приведены варианты реализации алгоритма с переходами от представле-

ния числа в прямом коде к представлению этого числа в обратном коде и от представления числа в обратном коде к его представлению в прямом коде.

On one algorithmic solution of the problem of restoring the number remaining in the residual class system

While performing operations expand the range of representations of numbers dividing, determining overflow, scaling, error correction calculations, there arises a problem of recovery of the remainder of the number on the module on the basis of the balance of this number for other modules of the system. Table recovery operation of the remainder of the number is implemented using the basic algorithm. Solution method based on the definition of residue on the module based on the balances on the remaining modules of the system. This definition of successive subtraction of constants from the obtained residue, and the aggregation of these constants to rezultes, which are formed on the module. At the same constant on each iteration are selected depending on the value of the residue in the analyzed discharge. The undoubted advantage of the stored requirements for the performance of the recovery operation, the remainder of the number. The aim of the study is an analytical review of the approach to accelerated implementation of the basic operations of recovery of the remainder of the number on the module on the basis of the balance of this number for other modules of the system. One implementation of the algorithm is its simultaneous implementation in the baseline for the desired number and number reverse search. This desired balance is determined by the value of the balance of the numbers for which the first result is a search result. Given the options the implementation of the algorithm with the transitions from the representation of a number to direct the code to the representation of that number in the reverse code and the representation of a number in the code back to its performance in direct code. The algorithm is implemented in the system of residual classes in the base the estimation of the recovery operation, the values of the balance number on the module based on the values of the residues that number for the rest of the system modules provides obtaining the desired result. Based on the proposed approaches is achieved by accelerating the implementation of the basic operations of recovery of the remainder of the number for this module. It seems appropriate to apply the proposed approaches as a promising research areas of this operation in the system of residual classes.

Полисский Юрий Давидович - старший научный сотрудник, Научно-исследовательский институт автоматизации черной металлургии, г.Днепр.

Поліський Юрій Давидович - старший науковий співробітник, Науково-дослідний інститут автоматизації чорної металургії, м.Дніпро.

Polissky Yuri - Older Researcher, Scientific Research Institute of Automation of Ferrous Metallurgy, Dnepr.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)