УДК 004

К. С. Исупов, А. А. Завиялов

ОБ ЭФФЕКТИВНОСТИ НОВОГО АЛГОРИТМА ВЫЧИСЛЕНИЯ РАНГА В СИСТЕМЕ ОСТАТОЧНЫХ КЛАССОВ*

Ранг числа – важная характеристика системы остаточных классов (СОК), показывающая, сколько раз диапазон системы был превышен при переходе от представления числа в СОК к его позиционному представлению. Знание ранга позволяет упростить оценку величины числа в СОК и выполнить ряд связанных операций, таких как определение знака и контроль переполнения диапазона. Поэтому эффективное нахождение ранга играет важную роль в практике применения СОК. В этой работе исследован новый алгоритм вычисления ранга, который не требует преобразования в двоичную систему счисления и предполагает выполнение только небольших целочисленных операций. Для *n*-модульной СОК алгоритм позволяет вычислить ранг за *n* итераций. На каждой итерации вычисления могут выполняться параллельно по модулям. Для нового алгоритма получены оценки быстродействия и затрат памяти.

Ключевые слова: система остаточных классов, ранг числа, алгоритмы, эффективность.

С ростом производительности суперкомпьютеров и объёмов производимых ими вычислений приобретают актуальность методы кодирования информации, допускающие эффективную параллельную обработку. Широкое распространение в этой области получили непозиционные системы счисления, а именно система остаточных классов [1, 5].

Система остаточных классов (СОК) – это система счисления, которая определяется набором взаимно простых целых чисел $m_1, m_2, ..., m_n$, называемых модулям. Динамический диапазон СОК представляется произведением модулей

**Работа выполнена при финансовой поддержке РФФИ в рамках научного проекта № 16-37-60003 мол_а_дк.*

$M = m_1 \times m_2 \times ... \times m_n$. Целое число X из интервала $[0, M-1]$ в СОК задаётся в виде вычетов по выбранным модулям:

$$X = \langle x_1, x_2, ..., x_n \rangle, \quad (1)$$

где $x_i \equiv X \bmod m_i$ для всех $i = 1, 2, ..., n$, то есть каждая цифра x_i – это наименьший положительный остаток от деления числа X на модуль m_i.

Прямое преобразование выполняется в соответствии с (1), а обратное определяется соотношением

$$X = \left| \sum_{i=1}^{n} x_i \, | M_i^{-1} |_{m_i} \, M_i \right|_M, \quad (2)$$

где M_i и $| M_i^{-1} |_{m_i}$ – константы, причем $M_i = M / m_i$, а $| M_i^{-1} |_{m_i}$ – мультипликативная инверсия M_i по отношению к m_i.

Все модульные операции над числами вида (1), например, сложение, вычитание без знака, умножение, выполняются параллельно над отдельными цифрами заданных чисел – остатки x_i не зависят друг от друга, вследствие этого такие операции могут выполняться без распространения переноса между разрядами. Это обусловливает возможность распараллеливания вычислений на уровне арифметических операций. Благодаря этой особенности СОК получила широкое распространение в таких направлениях высокоскоростной компьютерной арифметики, как цифровая обработка сигналов [2], криптография [3], обнаружение и исправление многократных ошибок кодирования [4], цифровая обработка изображений [6].

Немодульные операции, например, сравнение чисел и вычитание с отрицательным результатом, создают проблемы при работе с числами в СОК, поскольку сложность таких вычислений остаётся высокой. Для выполнения таких операций недостаточно знать лишь отдельные остатки x_i, необходима оценка значения числа X в целом. Но такую оценку затруднительно получить в виду непозиционной природы СОК.

Одной из характеристик позиционной величины числа в СОК, которая может быть использована для выполнения немодульных операций, является ранг числа. Согласно классическому определению [1], рангом числа *X* (функцией ранга) называется целое положительное число, показывающее, сколько раз диапазон системы *M* был превзойдён при переходе от представления *X* в СОК к его позиционному представлению с использованием (2). Ранг числа *X* будем обозначать $r(X)$. Эффективное вычисление ранга позволит во многих случаях ускорить процесс выполнения немодульных операций в СОК.

Следующая теорема из [1] позволяет определить ранг суммы двух чисел по известным рангам слагаемых.

Теорема (о ранге суммы). Если в системе с модулями $m_1, m_2, ..., m_n$ и диапазоном M заданы два числа, $X = \langle x_1, x_2, ..., x_n \rangle$ и $Y = \langle y_1, y_2, ..., y_n \rangle$, с рангами $r(X)$ и $r(Y)$ соответственно, то ранг $r(X+Y)$ суммы этих чисел определится как

$$r(X+Y) = r(X) + r(Y) - \sum_{i=1}^{n} \left(\left\lfloor \frac{x_i + y_i}{m_i} \right\rfloor \times \left| M_i^{-1} \right|_{m_i} \right). \qquad (3)$$

Теорема справедлива при условии невыхода суммы за модулярный диапазон. На основе этой теоремы в [1] предложен алгоритм, позволяющий определить ранг числа $X = \langle x_1, x_2, ..., x_n \rangle$ в СОК без перехода к его позиционному представлению. Он заключается в следующем.

Этап 1 (предвычислительный). На данном этапе вычисляются и сохраняются в ROM-память следующие числа в СОК вместе с их рангами:

$$\begin{aligned}
&T_1 = \langle 1, 1, ..., 1 \rangle, \\
&T_2 = \langle 0, m_1, m_1, ..., m_1 \rangle, \\
&\vdots \\
&T_i = \left\langle 0, ..., 0, \left| m_1 m_2 \cdots m_{i-1} \right|_{m_i}, ..., \left| m_1 m_2 \cdots m_{i-1} \right|_{m_n} \right\rangle, \\
&\vdots \\
&T_n = \left\langle 0, 0, ..., 0, \left| m_1 m_2 \cdots m_{n-1} \right|_{m_n} \right\rangle.
\end{aligned} \qquad (4)$$

Этап 2. Выполняются следующие шаги.

Шаг 1. К числу X прибавляется число T_1 столько раз, сколько потребуется для того, чтобы цифра числа X по модулю m_1, т.е. x_1, стала равной нулю. Пусть для этого потребовалось α_1 раз прибавить число T_1, и в результате суммирования получилось число X_1, то есть справедливо, что $X_1 = X + \alpha_1 T_1$. Тогда, применяя последовательно α_1 раз формулу (3), получим $r(X_1) = r(X) + \omega_1$, где ω_1 – известная величина.

Шаг 2. По аналогии с шагом 1, число T_2 прибавляется α_2 раз к X_1 до получения нулевого остатка по модулю m_2. В результате суммирования получается число X_2, то есть справедливо, что $X_2 = X_1 + \alpha_2 T_2$. При этом, $r(X_2) = r(X) + \omega_2$, где ω_2 – известная величина.

Шаг 3. Продолжая по индукции для оставшихся ненулевых цифр числа X по модулям $m_3, m_4, ..., m_n$, получим число в СОК $M = \langle 0,0,...,0 \rangle$, ранг которого равен –1. С другой стороны, ранг этого числа равен $r(X) + \omega_n$, где ω_n – известная величина. Отсюда следует, что искомый ранг числа X равен $r(X) = -\omega_n - 1$.

Данный алгоритм требует небольших затрат памяти и только малоразрядных целочисленных вычислений. Его недостатком является большое количество итераций, необходимых для последовательного обнуления всех цифр числа. В худшем случае число итераций равно $\sum_{i=1}^{n}(m_i - 1)$.

Рассмотрим модифицированный алгоритм вычисления ранга $r(x)$. Его основной идеей является безытерационное вычисление кратностей α_i и введение дополнительных подстановочных таблиц для хранения рангов $r(\alpha_i T_i)$. Пусть $X = \langle x_1, x_2, ..., x_n \rangle$ есть число, функцию ранга которого $r(X)$ необходимо вычислить. Модифицированный алгоритм состоит в следующем.

Этап 1 (предвычислительный). На данном этапе по-прежнему вычисляются остаточные коды (4), а также их мультипликативные инверсии

$|T_1^{-1}|_{m_1}, |T_2^{-1}|_{m_2}, ..., |T_n^{-1}|_{m_n}$, где $|T_i^{-1}|_{m_i}$ – такое целое положительное число от 1 до $m_i - 1$, что $T_i |T_i^{-1}|_{m_i} \equiv 1 \bmod m_i$ при $T_i = m_1 m_2 \cdots m_{i-1}$. Кроме этого, для каждого i-го модуля m_i вычисляется подстановочная таблица Tab_i размера $m_i - 1$ слов, каждая j-я строка которой $(1 \le j \le m_i - 1)$ содержит ранг $r(\alpha_i T_i)$, где $\alpha_i = \left| j \cdot |T_i^{-1}|_{m_i} \right|_{m_i}$. Также выделяется в памяти буфер размером n слов: $\omega = (\omega_1, \omega_2, ..., \omega_n)$, причем переменная ω_i соответствует i-му модулярному каналу, в котором производятся вычисления по модулю m_i.

Этап 2. Принимается обозначение $X_0 = X = \left\langle x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)} \right\rangle$ и выполняются следующие шаги.

Шаг 1. Вычисляется кратность $\alpha_1 = \left| (m_1 - x_1^{(0)}) \cdot |T_1^{-1}|_{m_1} \right|_{m_1}$.

Шаг 2. В СОК вычисляется произведение $\alpha_1 T_1 = \left\langle t_1^{(1)}, t_2^{(1)}, ..., t_n^{(1)} \right\rangle$ и далее сумма $X_1 = X_0 + \alpha_1 T_1$, которая будет иметь вид $X_1 = \left\langle 0, x_2^{(1)}, \ldots, x_n^{(1)} \right\rangle$. Если при суммировании был переход через модуль m_i, то это фиксируется в переменной ω_i, то есть: $\omega_1 = 1, \omega_2 = \left\lfloor \frac{x_2^{(0)} + t_2^{(1)}}{m_2} \right\rfloor, \ldots, \omega_n = \left\lfloor \frac{x_n^{(0)} + t_n^{(1)}}{m_n} \right\rfloor$. Из $(m_1 - x_1^{(0)})$-й строки подстановочной таблицы Tab_1 выбирается ранг $r(\alpha_1 T_1)$.

Шаг 3. В СОК вычисляется произведение $\alpha_2 T_2 = \left\langle 0, t_2^{(2)}, ..., t_n^{(2)} \right\rangle$ и сумма $X_2 = X_1 + \alpha_2 T_2$, которая будет иметь вид $X_2 = \left\langle 0, 0, x_3^{(2)}, \ldots, x_n^{(2)} \right\rangle$. Если при суммировании был переход через модуль m_i, то это фиксируется в переменной ω_i, то есть: $\omega_2 = \omega_2 + 1$ и $\omega_i = \omega_i + \left\lfloor \frac{x_i^{(1)} + t_i^{(2)}}{m_i} \right\rfloor$ для всех $i = 3, 4, ..., n$. Из $(m_2 - x_2^{(1)})$-й строки подстановочной таблицы Tab_2 выбирается ранг $r(\alpha_2 T_2)$.

Шаг 4. Описанный процесс повторяется для всех $i = 3, 4, ..., n$. В конечном счете получается число $X_n = M = \langle 0, 0, ..., 0 \rangle$, функция ранга которого равна −1. С другой стороны, в соответствии с теоремой о ранге суммы,

$r(X_n) = r(X) + \sum_{i=1}^{n} r(\alpha_i T_i) - \sum_{i=1}^{n} \omega_i |M_i^{-1}|_{m_i}$. Отсюда получается уравнение для искомой функции ранга $r(X)$:

$$r(X) = \sum_{i=1}^{n} \omega_i |M_i^{-1}|_{m_i} - \sum_{i=1}^{n} r(\alpha_i T_i) - 1.$$

Представленный алгоритм также требует только малоразрядных целочисленных операций и позволяет вычислить ранг числа за n итераций, где n – число модулей СОК.

На рис. 1 представлены результаты экспериментов по сравнению времени выполнения классического и нового алгоритмов расчёта ранга числа СОК. Новый алгоритм значительно быстрее, чем классический вариант, причём с увеличением количества модулей разница становится более существенной. Например, при использовании СОК из 5-10 модулей время работы классического алгоритма больше времени работы нового алгоритма на 20–100%, а при использовании 60–64 модулей – на 1500-2000%.

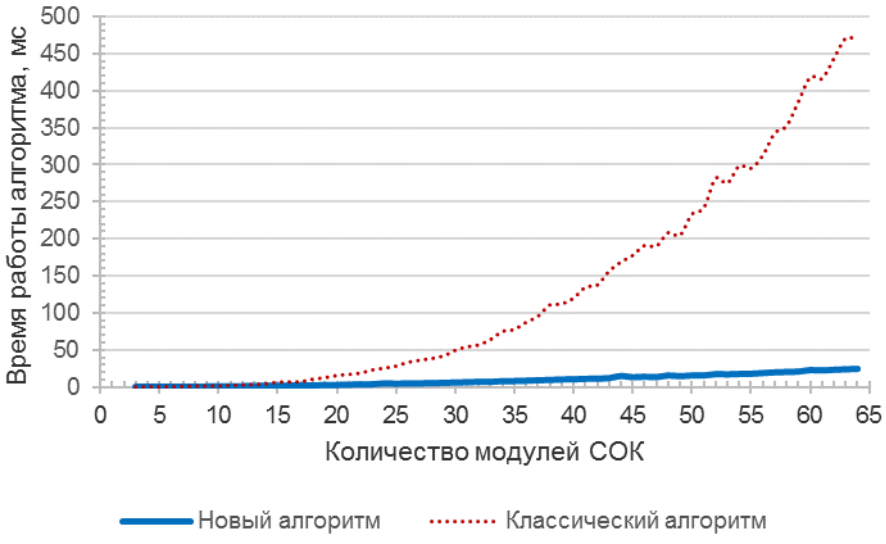

Рис. 1. Время работы классического и нового алгоритмов расчёта ранга в СОК

На рис. 2 показано время выполнения нового алгоритма и многоразрядного алгоритма, в котором для расчёта ранга используется перевод числа из СОК

в позиционное представление с использованием библиотеки длинной арифметики The GNU MP Bignum Library.

Рис. 2. Время работы нового и многоразрядного алгоритмов расчёта ранга

Время выполнения нового алгоритма сравнимо со временем многоразрядного алгоритма, который выполняется с использованием оптимизированной библиотеки. При реализации нового алгоритма не выполнялось никаких специальных оптимизаций программного кода. Тем не менее, при увеличении количества модулей до 60–64 время работы многоразрядного алгоритма на 10–20% больше времени работы нового алгоритма и с ростом числа модулей выигрыш нового алгоритма становится более существенным.

На рис. 3 представлены результаты оценки классического и нового алгоритмов по потребляемой памяти. Новый алгоритм потребляет больше памяти, поскольку помимо минимальных чисел T_i необходимо хранить их мультипликативные инверсии и подстановочные таблицы Tab_i.

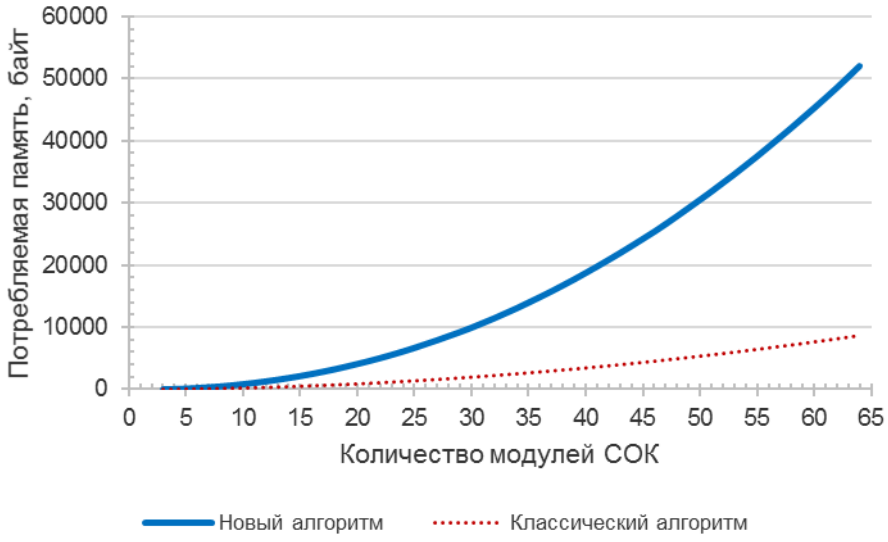

Рис. 3. Затраты памяти классического и нового алгоритмов

Таким образом, новый алгоритм вычисления ранга числа в СОК обладает высоким быстродействием при приемлемых затратах памяти и может быть использован для ускорения ряда немодульных процедур, таких как сравнение. Алгоритм не требует привлечения многоразрядных библиотек, поэтому может быть легко реализован на многих вычислительных платформах, включая графические ускорители, ПЛИС и ASIC. Дополнительное ускорение может быть получено при распараллеливании вычислений по модулям на каждой итерации.

Список литературы

1. *Акушский И. Я., Юдицкий Д. И.* Машинная арифметика в остаточных классах. М.: Сов. Радио, 1968. 440 с.

2. *Albicocco P., Cardarilli G., Nannarelli A., Re M.* Twenty years of research on RNS for DSP: Lessons learned and future perspectives // Proceedings of 14th Int. Symp. Integrated Circuits (ISIC). Singapore. 2014. P. 436–439.

3. *Esmaeildoust M., Schinianakis D., Javashi H., Stouraitis T., Navi K.* Efficient RNS implementation of elliptic curve point multiplication over GF(p) // IEEE Transactions on VLSI Systems. 2013. Vol. 21. № 8. P. 1545–1549.

4. *Goh V. T. and Siddiqi M. U.* Multiple error detection and correction based on redundant residue number systems // IEEE Transactions on Communications. 2008. Vol. 56. № 3. P. 325–330.

5. *Omondi A.* Residue Number Systems: Theory and Implementation. L.: Imperial College Press, 2007. 312 p.

6. *Wang W., Swamy M., Ahmad M.* RNS application for digital image processing // Proceedings of 4th IEEE Int. Workshop Syst.-on-Chip for Real Time Applications. Banff, Alberta, Canada, 2004. P. 77–80.

ИСУПОВ Константин Сергеевич – кандидат технических наук, ведущий научный сотрудник кафедры электронных вычислительных машин, Вятский государственный университет. 610000, г. Киров, ул. Московская, 36.

E-mail: isupov.k@gmail.com

ЗАВИЯЛОВ Антон Андреевич – студент кафедры электронных вычислительных машин, Вятский государственный университет. 610000, г. Киров, ул. Московская, 36.

E-mail: antonzaviyalov@gmail.com