УДК 681

И. В. Степанов, В. С. Ростовцев

ПРИНЦИПЫ И АЛГОРИТМЫ РАЗРАБОТКИ ИНСТРУМЕНТАЛЬНОЙ ПРОГРАММЫ ОЦЕНКИ ЭФФЕКТИВНОСТИ МОДУЛЯРНОЙ АРИФМЕТИКИ

Рассматриваются принципы и алгоритмы разработки инструментальной программы оценки эффективности модулярной арифметики, реализующей вычисления операций сложения, вычитания, умножения и деления в системе остаточных классов (СОК). В настоящее время не существует методов оценки эффективности решения вычислительной задачи в системе остаточных классов в отведенный лимит времени при заданной точности. Решение вычислительных задач на базе позиционной системы счисления (ПСС) сопровождается значительными временными затратами при установленной точности вычисления [1]. Система остаточных классов в свою очередь позволяет решить эту проблему, за счет перехода к обработке малоразрядных данных, что дает возможность организовать параллельные вычисления на уровне элементарных арифметических операций. Предлагаются методы и алгоритмы программной реализации перевода чисел из классической ПСС в СОК и обратно на основе Китайской теоремы об остатках. Представлены форматы и характеристики модулярных чисел, в соответствии с которыми в СОК выполняются модульные и немодульные операции. В результате получено описание функционирования инструментальной программы оценки эффективности модулярной арифметики и области ее применения.

Ключевые слова: система остаточных классов, модулярная арифметика, позиционная система счисления, китайская теорема об остатках, интервально-позиционная характеристика.

Вопросам применения системы остаточных классов для решения вычислительных задач посвящено много публикаций [1–5,7]. Однако не существует методики оценки временных затрат для заданного набора арифметических операций в СОК. Предлагаемая непозиционная система счисления характеризуется естественным внутренним параллелизмом, поскольку переход к

обработке малоразрядных данных позволяет использовать параллельные вычисления на уровне элементарных арифметических операций [1]. Отличительной особенностью модулярной арифметики является отсутствие переносов между соседними цифрами чисел, что не только повышает быстродействие, но и позволяет эффективно распределять и векторизовать вычисления, поскольку нет зависимости по данным. Одновременно с этим модулярная арифметика обеспечивает высокую устойчивость к ошибкам округления. Применение СОК на сегодняшний день – актуальная задача, поскольку параллелизм данных, который она позволяет реализовать может повысить быстродействие решения многих задач таких как вычисление рядов, решение систем уравнений больших порядков, интерполяция, расчеты и моделирование в химии, гидродинамике, ядерной физике и другие [1].

Для оценки временной эффективности решения прикладных задач в СОК предлагается использовать инструментальную программу, с помощью которой можно оценить временные затраты на выполнение «смеси» арифметических операций сложения, вычитания, умножения, деления в системе остаточных классов для форматов с фиксированной и плавающей точкой при заданной разрядности. Например, «смесь» включает: 723 операции сложения, 118 операций вычитания, 1198 операций умножения и 423 операций деления.

В результате исполнения программы для заданной пользователем «смеси» операций в СОК для заданной разрядности и формата представления чисел (плавающая или фиксированная точка) можно оценить следующие характеристики быстродействия по сравнению с ПСС:

1. Процент временных затрат на преобразование чисел из ПСС в СОК.
2. Процент временных затрат на выполнение операций заданной «смеси» и отдельно по каждому типу операций.
3. Процент временных затрат на преобразование чисел из СОК в ПСС.

4. Оценка временной эффективности выполнения заданной «смеси» в СОК в сравнении с выполнением этой «смеси» в ПСС при выбранной пользователем разрядности чисел.

Для разработки программы выбран на язык C/C++ с применением библиотек высокой точности (GMP, MPFR) для представления больших чисел в позиционной системе счисления [6].

Для любой системы [1,5] взаимно простых чисел (p_1, p_2,…, p_n), любое число X из диапазона [0; M) взаимно-однозначно представимо в виде вектора (a_1, a_2, …, a_n), где

$a_i = X \bmod p_i$;

mod – операция вычисления остатка от целочисленного деления X на p_i;

$M = p_1*p_2*\ldots*p_n$ – предел представимых чисел;

p_1, p_2,…, p_n – модули системы (основание системы остаточных классов);

a_1, a_2, …, a_n – остатки (вычеты) числа по заданной системе модулей.

Поскольку в системе остаточных классов, в отличии от позиционной системы, вычисления по каждому разряду (вычету) производятся независимо от остальных, то можно организовать параллельное выполнение по N каналам, что позволит многопоточному приложению выполнять операции с длинной арифметикой в N раз быстрее.

Систему используемых модулей подбирают под конкретную задачу. Для представления 32-х битных чисел достаточно следующей системы модулей: (7, 11, 13, 17, 19, 23, 29, 31) – все они взаимно простые числа друг с другом, их произведение равно 6685349671, что превышает значение 4294967296 (2^{32}). Каждый из модулей не превышает 5 бит, то есть операции сложения и умножения могут параллельно производиться над 5-битными числами (на 27 разрядов меньше).

Прямое преобразование из ПСС (обычно в двоичном виде) в систему счисления в остатках заключается в нахождении остатков от деления по каждому из модулей системы.

Для обратного преобразования числа из системы остаточных классов в ПСС существует метод на базе Китайской теоремы об остатках[1] или системы ортогональных базисов и метод на базе полиадического кода (система, со смешанным основанием).

Способ, основанный на китайской теореме об остатках[1], базируется на следующем представлении числа[5]:

$X = (x_1, x_2,\ldots, x_n) = (x_1,0,\ldots,0) + (0, x_2,\ldots,0) + \ldots + (0, 0,\ldots, x_n) =$

$= x_1 * (1,0,\ldots,0) + x_2 * (0,1,\ldots,0) + \ldots + x_n * (0,0,\ldots,1).$ (1)

Таким образом, для обратного преобразования требуется найти систему ортогональных базисов: $B_1 = (1,0,\ldots,0)$, $B_2 = (0,1,\ldots,0)$, … , $B_n = (0,0,\ldots,1)$.

Эти вектора находятся один раз для заданного базиса, а для их поиска требуется решить уравнение вида:

$(M_i * b_i) \bmod p_i = 1$, где

$M_i = M/p_i$, а b_i – искомое число.

b_i является мультипликативной инверсией M_i и может быть обозначено как $\left|M_i^{-1}\right|_{pi}$ или $M_i^{-1} \bmod pi$.

В этом случае позиционное представление, следующее:

$B_i = M_i * b_i$

$X = [x_1 * (M_1 * b_1) + x_2 * (M_2 * b_2) + \ldots + x_n * (M_n * b_n)] \bmod M.$ (2)

В системе остаточных классов определены основные арифметические операции, которые можно условно поделить на две группы:

1. Модульные операции, к которым относятся: арифметическое сложение и вычитание модулярных чисел, умножение модулярных чисел, поразрядное сравнение модулярных чисел и другие.

2. Немодульные операции, которые включают: сравнение, деление, контроль переполнения, округление модулярных чисел, масштабирование и другие [7].

Немодульные операции задействованы в выполнении всех арифметических операций над числами с плавающей точкой, поэтому необходимо определить

методы их ускоренного выполнения. Они требуют дополнительных накладных расходов, поскольку невозможно явно определить результат операции по значению разрядов модульного числа.

В основе ускоренных алгоритмов выполнения немодульных операций лежат методы вычислений приближенных позиционных характеристик [4] модулярных чисел, поскольку они позволяют получить позиционный эквивалент модулярного числа с более низкой вычислительной сложностью, чем вычисление точных характеристик.

Уточненная приближенная позиционная характеристика, в соответствии с китайской теоремой об остатках, может быть представлена следующим образом [3]:

$$C_x = \frac{X}{M} \approx \left|\sum_{i=1}^{n} K_i x_i\right|_1, \text{ где} \qquad (3)$$

$X = (x_1, x_2, \ldots, x_n)$ – модулярное число, определяемое модулярными разрядами $(x_1, \ldots, x_n)$;

x_i, $i=1,\ldots,n$ – модулярные разряды числа;

$M = p_1 * p_2 * \ldots * p_n$ – предел представимых чисел;

$p_1, p_2, \ldots, p_n$ – модули системы (основание системы);

$|\ \ |_1$ – дробная часть аргумента;

$K \approx \frac{|M_i^{-1}|_{p_i}}{p_i}$ – константы, вычисляемые заранее и округленные в пределах допустимой разрядности модульных разрядов;

$\left|M_i^{-1}\right|_{p_i}$ – мультипликативная инверсия от M_i;

$M_i = \frac{M}{p_i}$;

$|x|_n$ – операция x mod n.

Временная сложность немодульных операций на базе приближенной позиционной характеристики при последовательном выполнении O (n), при

параллельном O (log n), что на порядок ниже, чем методы на основе смешанной системы с использованием ПСС.

Позиционная характеристика числа не исключает критических ошибок округления при большом числе модулей СОК. Для решения этой проблемы необходимо использовать метод выполнения немодульных операций на основании интервально-позиционных величин [2,3].

Интервально-позиционная характеристика $I(\frac{X}{M})$ – это отрезок с направленно округленными позиционными границами, которые удовлетворяют условию:

$$\downarrow \frac{X}{M} \leq E(\frac{X}{M}) \leq \uparrow \frac{X}{M}, \text{ где} \qquad (4)$$

$E(\frac{X}{M})$ – точное значение позиционной характеристики;

$\downarrow \frac{X}{M}$– значение позиционной характеристики с округлением по недостатку;

$\uparrow \frac{X}{M}$ – значение позиционной характеристики с округлением по избытку.

Таким образом, интервально-позиционная характеристика (ИПХ) отображает величину модулярного числа на диапазон [0,1). Схематически это отображение представлено на рисунке 1[2].

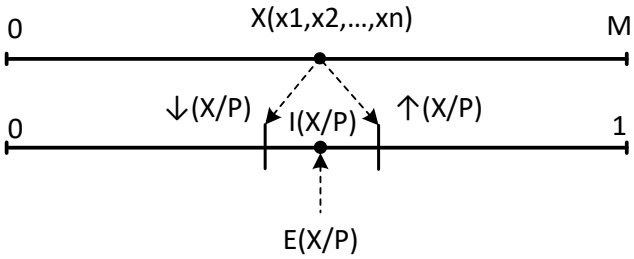

Рис. 1. Интервально-позиционная характеристика числа

На основе анализа границ ИПХ строится оценка величины каждого числа и выполняется операция сравнения. Алгоритм сравнения модульных чисел X = (x_1,x_2 ,…, x_n) и Y = (y_1, y_2 ,…, y_n) в этом случае состоит из следующих шагов:

1. Расчет интервально-позиционной характеристики $I(\frac{Y}{M})$;

2. Определение условия соблюдения корректности результата;

3. Определение результата исходя из условий: $\downarrow \frac{X}{M} > \uparrow \frac{Y}{M} \rightarrow X > Y$; $\uparrow \frac{X}{M} < \downarrow \frac{Y}{M} \rightarrow X < Y$; $\square\left(\frac{\square}{\square}\right) = \square\left(\frac{\square}{\square}\right) < 0 \rightarrow \square = \square$.

Для организации быстродействующих высокоточных вычислений в системе остаточных классов предусматривается формат с плавающей точкой для представления чисел большой разрядности[2]:

$x \rightarrow \left\{\square, \square, \square, \square\left(\frac{\square}{\square}\right)\right\}$, где (5)

X – мантисса модулярного числа в диапазоне от [0; M-1)

$\square\left(\frac{\square}{\square}\right)$ – ИПХ мантиссы.

s – знак числа;

$\square$ – порядок со знаком;

M – произведение модулей.

Применимая структура данных для представления модулярного числа в формате с плавающей точкой представлена в таблице 1.

Таблица 1

Формат представления модулярного числа

sign	exponent	IPC1	IPC2	mantissa[1]	…	mantissa[n]

Поля имеют следующие значения: sign – знак; exponent – смещенный порядок, представленный аналогичное спецификации IEEE-754 [6]; IPC1 – нижняя граница ИПХ; IPC2 – верхняя граница ИПХ; mantissa[n] – массив вычетов модулярного числа.

В результате для нахождения суммы или разности чисел с плавающей запятой необходимо выполнить следующие шаги:

1. Анализ флагов is_null (незначимое число), is_infinity (обозначение бесконечности), is_nan (исключительная ситуация), если они равны 0, то переход к пункту 2, иначе числа обрабатывается в соответствии со стандартом IEEE – 754 [6].

2. Находится разность порядков. Если разность порядков больше размерности входных данных, то результатом суммы станет число с большим порядком, иначе к шагу 3.

3. Производится выравнивание порядков. Результат имеет порядок числа с большим порядком.

4. Если числа одного знака, то производится сумма (вычитание) мантисс и производится округление, иначе производится разность (сложение) мантисс и определяется знак результата.

Для нахождения суммы (разности) чисел с фиксированной запятой необходимо выполнить только 4 пункт.

Знак результата определяется с помощью анализа величин операндов на основе интервально-позиционной характеристики.

Для нахождения произведения чисел с плавающей запятой необходимо выполнить следующие шаги:

1. Анализ флагов is_null, is_infinity, is_nan, если они равны 0, то переход к пункту 2, иначе числа обрабатывается в соответствии со стандартом IEEE – 754 [6].

2. Порядок результата равен сумме порядков множителей.

3. Определение знака результата на основе знаков множителей.

4. Мантисса результата равна произведению мантисс множителей.

5. Округление результата.

Для нахождения произведения чисел с фиксированной запятой необходимо выполнить только 3 и 4 пункты.

С помощью инструментальной программы пользователь может задать формата представления чисел (плавающая или фиксированная точка), разрядность чисел, параметры «смеси» (количество арифметических операций каждого типа) и вводимые исходные данные. На основании размерности введенных данных автоматически формируется система модулей СОК. Далее вычисляется ИПХ и выходной массив модулярных мантисс, с помощью чего получаем модулярное представление исходного числа в модуле преобразования ПСС в СОК. Далее

выполняется заданные пользователем «смеси» арифметических операций. По возможности, вычисления в СОК выполняются параллельно. На каждом этапе алгоритма регистрируется протокол выполнения. После преобразования данных из СОК в ПСС осуществляется визуализация протокола выполнения всех этапов вычислений и преобразований из одной системы счисления в другую.

В результате реализации инструментальной программы оценки эффективности модулярной арифметики для чисел в формате с фиксированной и плавающей запятой пользователю будет предоставлен инструмент оценки временной эффективности выполнения «смеси» арифметических операций для различной разрядности чисел и форматов представления чисел.

Ожидаемое ускорение вычислений в СОК по сравнению с аналогичным применением библиотек высокоточных вычислений на базе позиционных систем счисления будет документально зарегистрировано программой и может служить основанием необходимости и целесообразности реализации этой «смеси» арифметических операций на ПЛИС [7].

Список литературы

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

2. *Исупов К. С.* Методы и алгоритмы организации высокоточных вычислений в арифметике остаточных классов для универсальных процессорных платформ: автореф. дис. … канд. техн. наук. Пенза, 2014.

3. *Князьков В. С., Исупов К. С.* Уточненное вычисление приближенной позиционной характеристики чисел в системе остаточных классов. Киров: Вят. гос. ун-т, 2010.

4. Приближенный метод выполнения немодульных операций в системе остаточных классов / Н. И. Червяков, В. М. Авербух, М. Г. Бабенко, П. А. Ляхов, А. В. Гладков, А. В. Гапочкин // Фундаментальные исследования. 2012. № 6. С. 189–193.

5. Сайт Habrahabr.ru. Введение в модулярную арифметику. URL: http://habrahabr.ru/post/144886/, свободный.

6. 754-2008 – IEEE Standard for Floating-Point Arithmetic (ANSI/IEEE Std 754-1985) – The Institute of Electrical and Electronics Engineers, Inc 345 East 47th Street, New York, NY 10017, 2008, USA

7. *Сучок С., Ростовцев В. С.* Организация деления чисел в системе остаточных классов на базе ПЛИС // Общество, наука, инновации: НПК-2016. Киров, 2016. С. 2193–2197.

СТЕПАНОВ Иван Вячеславович – студент IV курса, направление подготовки бакалавра «Информатика и вычислительная техника», Вятский государственный университет. 610000, г. Киров, ул. Московская, 36.

E-mail: youshallnotpass213@gmail.com

РОСТОВЦЕВ Владимир Сергеевич – кандидат технических наук, доцент кафедры электронных вычислительных машин, Вятский государственный университет. 610000, г. Киров, ул. Московская, 36.

E-mail: rostov_kirov@mail.ru