УДК 004.42

МНОГОПОТОЧНАЯ JAVA-РЕАЛИЗАЦИЯ ДВУХТОЧЕЧНОГО БЛОЧНОГО ОДНОШАГОВОГО МЕТОДА ИНТЕГРИРОВАНИЯ

Кудерметов Равиль, Польская Ольга

Запорожский национальный технический университет

Аннотация

В данной работе рассмотрена многопоточная реализация двухточечного блочного одношагового метода интегрирования обыкновенных дифференциальных уравнений с использованием механизма блокирующей очереди, предоставляемого библиотекой языка Java. Приведены численные результаты для иллюстрации эффективности такой реализации.

Abstract

In this paper the multithreading implementation of the two-point block one-step method for solving ordinary differential equations is considered. For this purpose the blocking queue mechanism of Java's LinkedBlockingQueue library class was used. Numerical results are given to illustrate the effectiveness of this implementation.

Введение

В настоящее время одним из основных направлений повышения производительности компьютеров является развитие технологий многоядерных процессоров. Двух- и четырех-ядерные процессоры используются в большинстве современных компьютеров, что значительно ускоряет выполнение как обычных, так и научных приложений. Эффективность использования многоядерных процессоров для научных расчетов и моделирования можно значительно повысить, если создаваемые программные приложения будут по своей архитектуре и применяемым методам соответствовать архитектуре многоядерных процессоров [1].

При моделировании динамических систем значительную часть составляют задачи, в которых необходимо интегрировать системы обыкновенных дифференциальных уравнений. Распространенной практикой является применение для этих целей численных методов 4-го порядка точности, например методов Рунге-Кутты. Такую и более высокую точность могут обеспечить двухточечные и четырехточечные блочные одношаговые и многошаговые методы интегрирования, в которых вычисления можно проводить независимо и одновременно в двух или четырех точках, соответственно. Очевидно, напрашивается вывод о соответствии этих методов архитектурам двух- и четырех-ядерных процессоров и о возможности и необходимости дальнейшего совершенствования этих методов, особенно с целью более широкого их применения в практических приложениях.

Учитывая широкое использование технологии Java при создании современных приложений и развитые средства многопоточности этой технологии, в данной работе рассмотрены некоторые особенности реализации двухточечного блочного одношагового метода интегрирования с использованием потоков Java и их отражение на двух-ядерные процессоры, а также приведены некоторые результаты экспериментальной проверки реализации.

Параллельный алгоритм метода и его многопоточная реализация

Численное решение задачи Коши для обыкновенных дифференциальных уравнений

$$\frac{dy}{dt} = f(t, y),\ y(t_0) = y_0 \qquad (1)$$

на равномерной сетке по аргументу t с шагом h двухточечным (k = 2) блочным одношаговым методом интегрирования реализуется следующей совокупностью рекуррентных формул [2]:

$$y_{n+k,0} = y_n + khF_n,\ k = 1, 2, \tag{2}$$

$$y_{n+1,s+1} = y_n + \frac{h}{12}\left(5F_n + 8F_{n+1,s} - F_{n+2,s}\right), \tag{3}$$

$$y_{n+2,s+1} = y_n + \frac{h}{3}\left(F_n + 4F_{n+1,s} + F_{n+2,s}\right),\ s = \overline{0,2}, \tag{4}$$

где y_n - численная аппроксимация точного решения (1) в точке n сетки по аргументу t;
$y_{n+k,s}$ - численная аппроксимация точного решения (1) в k-й точке блока на итерации s;
$F_n = f(t_n, y_n)$ - значение функции $f(t, y)$ правой части (1) в точке n;
$F_{n+k,s+1} = f(t_n + kh, y_{n+k,s})$ - значение функции $f(t, y)$ в k-й точке блока на итерации s.

Организовать итерации по формулам (3) и (4) можно с помощью параллельно работающих на различных ядрах процессов или потоков многоядерного процессора, либо на разных процессорах многопроцессорной системы. В данной работе рассматривается реализация приведенного алгоритма с использованием технологии Java, которая имеет множество эффективных инструментов многопоточного программирования.

На рис. 1 схематично показана реализация параллельного двухточечного блочного одношагового метода интегрирования с использованием одного из вариантов блокирующей очереди - класса `LinkedBlockingQueue`, содержащегося в пакете `java.util.concurent`. С помощью методов `put()` и `take()` этого класса потоки обмениваются между собой результатами каждой итерации через блокирующие очереди, и за счет блокировок доступа к ним обеспечивается взаимная синхронизация потоков.

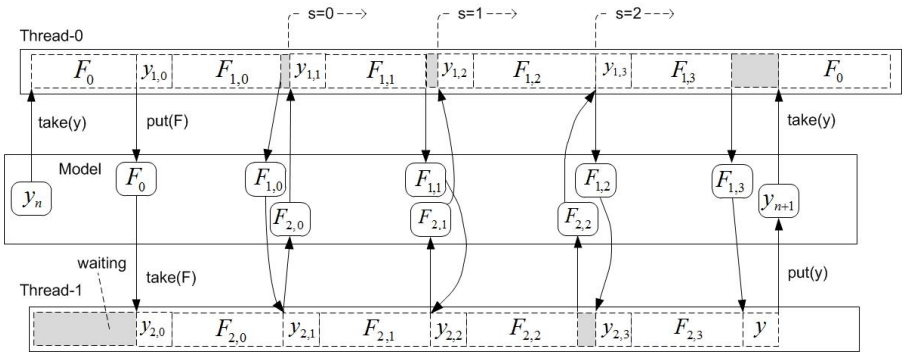

Рисунок 1 – Схема многопоточной реализации метода

В работе [3] получены оценки точности блочных одношаговых и многошаговых методов интегрирования. Для одношаговых методов локальная ошибка в узлах блока будет иметь порядок $O\left(h^{k+2}\right)$ после выполнения $k+1$ шагов по формулам (3) и (4). При этом шаг интегрирования необходимо выбирать согласно условию

$$h < \frac{1}{kL\|A\|}, \tag{5}$$

где L - константа из условия Липшица $\left|f(t, y_1) - f(t, y_2)\right| \le L\left|y_1 - y_2\right|$, которому должна удовлетворять функция правой части уравнения (1) при любых t, y_1, y_2 из рассматриваемой области;

$\|A\| = 0{,}833$ - норма матрицы коэффициентов перед функциями $F_{n+1,s}$ и $F_{n+2,s}$ в формулах (3) и (4). Элементы матрицы A для двухточечного блочного одношагового метода имеют следующие значения:

$$A = \begin{pmatrix} \frac{2}{3} & -\frac{1}{12} \\ \frac{2}{3} & \frac{1}{6} \end{pmatrix}. \tag{6}$$

Для тестирования правильности программной реализации можно использовать, например, умеренно устойчивую задачу [3]

$$\frac{dy}{dt} = -10(t-1)y,\ y(0) = 1,\ t = [0, 2], \tag{7}$$

имеющую аналитическое решение $y(t) = \exp(-5t(t-2))$. Максимальная ошибка при шаге интегрирования h = 0,01 будет не более $1{,}37 \cdot 10^{-4}$ в точке t = 1.

На практике максимальные эффективность или ускорение метода можно достичь, если временные затраты на вычисление правой части (1) уравнения значительно превосходят время, затрачиваемое на реализацию формул (2)-(4) и обеспечение синхронизации потоков. Теоретическое ускорение можно оценить как отношение количеств вычислений функции правой части (1) при последовательной и параллельной реализациях. Очевидно (см. рис. 1), оно равно 9/5, т. е. 1,8. На рис. 2 приведены результаты экспериментальной оценки ускорения многопоточной реализации двухточечного блочного одношагового метода в зависимости от доли временных затрат на вычисление правой части уравнения (1), полученные с использованием двух-ядерного процессора Intel® Core™ i3-2310M @ 2.10 GHz.

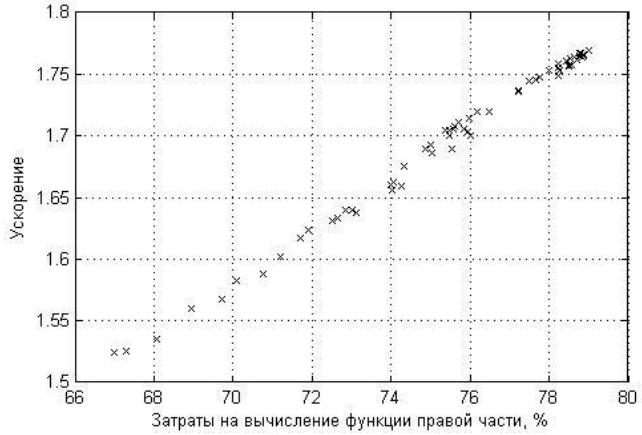

Рисунок 2 – Зависимость ускорения метода от доли затрат времени на вычисление функции правой части уравнения (1)

Список использованных источников:

1. Эхтер Ш. Многоядерное программирование / Ш. Эхтер, Дж. Робертс. – СПб.: Питер, 2010. – 316 с. - ISBN 978-5-388-00091-0.

2. Системы параллельной обработки / Под ред. Д. Ивенса. – М.: Мир,1985. – 416 с.

3. Фельдман Л. П. Эффективные методы распараллеливания численного решения задачи Коши для обыкновенных дифференциальных

уравнений / Л. П. Фельдман, О. А. Дмитриева. // Математическое моделирование, 13:7 (2001), С. 66-72.