УДК 681.5

ПАРАЛЛЕЛЬНЫЕ АЛГОРИТМЫ ЧИСЛЕННОГО РЕШЕНИЯ ЗАДАЧИ КОШИ ДЛЯ СОДУ

Назарова И.А.
Донецкий национальный технический университет

Запропоновано паралельні чисельні алгоритми однокрокових методів для вирішення задачі Коші. Розроблені обчислювальні схеми відображення методів на паралельні структури різної топології: лінійка/кільце, сітка/тор, гіперкуб. Отримані порівняльні характеристики потенційного та реального паралелізму, проведені чисельні експерименти на системі тестів.

В докладе рассматриваются параллельные алгоритмы численного решения нелинейной задачи Коши для СОДУ на основе явных (ЯМРК), полностью неявных (ПНМРК) и гибридных одношаговых методов с альтернативными способами оценки апостериорной локальной погрешности, а также особенности применения этих методов для решения линейных задач. Предлагаемые алгоритмы ориентированы на использование в многопроцессорных вычислительных системах SIMD или MIMD архитектуры с различными топологиями процессорных элементов.

Численно решается задача Коши для системы обыкновенных дифференциальных в общем случае нелинейных уравнений (СОДУ) первого порядка размерности m с известными начальными условиями:

$$\begin{cases} \bar{y}' = F(x, \bar{y}) \\ \bar{y}(x_0) = \bar{y}_0 \end{cases}, \qquad (1)$$

где $\bar{y} = (y_1, y_2, ..., y_m)^T$, $\bar{y}_0 = (y_{10}, y_{20}, ..., y_{m0})^T$, $F = (f_1, f_2, ..., f_m)^T$.

Одним из главных вопросов, возникающих при численном решении систем обыкновенных дифференциальных уравнений, является проблема оценки погрешности приближенного решения. Априорная оценка глобальной погрешности разностного метода позволяет судить о сходимости приближенного решения задачи к точному и, следовательно, о его применимости. Апостериорная оценка локальной погрешности, получаемая на каждом шаге

вычислений, позволяет автоматически выбирать шаг интегрирования, обеспечивающий заданную точность приближенного решения. Оценки шаговой погрешности решения осуществлялись на основе следующих технологий: дублирование шага; использование вложенных методов на основе явной и неявной разностных схем; технология локальной экстраполяции Ричардсона. Полуэмпирическое правило Рунге как для последовательных, так и в случае параллельных алгоритмов обладает высокой вычислительной сложностью. Экстраполяционная технология Ричардсона включает численный метод решения задачи Коши, последовательность сеток, рекуррентное правило вычисления значений приближенного решения. Эффективность применения технологии локальной экстраполяции напрямую зависит от правильного выбора и сочетания всех трех составляющих этого метода. В качестве опорных алгоритмов при параллельной реализации использовались явные и неявные методы Рунге-Кутты. Величина шага интегрирования инициировалась на основе четных последовательностей, допускающих для симметричных методов разложение по степеням h^2.

Для разработки параллельных алгоритмов используется многоэтапный технологический процесс - иерархическая декомпозиционная методика в сочетании с математическим аппаратом графов влияния [2].

Для параллельных вычислений важным, а, иногда и определяющим, является учет сложности межпроцессорных связей. Поскольку для параллельных методов решения задачи Коши информационное взаимодействие по типу коммуникаций является структурным, предоставляется возможность разработать единые коммуникационные примитивы для различных операций передачи данных в различных топологиях. Наиболее используемые топологии: линейка/кольцо, решетка/тор, гиперкуб и две коммуникационные операции: передача данных между двумя процессорами сети – операция типа “*point-point*” и передача данных от всех процессоров сети всем процесорам сети – множественная пересылка “*all-to-all*”.

Для оценки времени выполнения операции передачи одного сообщения объемом V байт между двумя процессами локализованными на различных процессорах при распределенной памяти использовалась следующая модель:

$$T_{p-p} = t_s + t_w \cdot V \cdot l,\ t_w = \frac{y}{B}, \qquad (2)$$

где t_s – латентность, длительность подготовки сообщения для передачи; l – длина маршрута; t_w – время передачи одного байта; y – число байт в слове; B – пропускная способность канала передачи данных (байт/секунда). Трудоемкость одиночной операции пересылки данных между двумя процесорами может быть получена путем подстановки длины диаметра сети в выражение (2). Для вычисления времени выполнения множественной пересылки в условиях той же модели необходимо определиться с выбором алгоритма маршрутизации. К числу наиболее распространенных оптимальных алгоритмов передачи данных относится класс методов покоординатной маршрутизации. Идея этих методов заключается в том, что поиск путей передачи данных осуществляется последовательно для каждой размерности рассматриваемой топологии. Сокращение времени передачи данных может быть достигнуто и за счет использования встречных обменов, когда данные передаются одновременно в обе стороны. Это позволяет добиться почти двойного уменьшения времени обмена данными по сравнению с однонаправленными обменами. Таким образом, в условиях существования двунаправленных линков имеем:

– для кольцевой топологии:

$$T^R_{p-p} = t_s + V \cdot t_w \cdot \lfloor p/2 \rfloor,$$
$$T^R_{all-to-all} = (t_s + V \cdot t_w) \cdot (p-1),$$

– для топологии решетка:

$$T^M_{p-p} = t_s + 2V \cdot t_w \cdot \lfloor \sqrt{p}/2 \rfloor,$$
$$T^M_{all-to-all} = 2t_s(\sqrt{p}-1) + V \cdot t_w \cdot (p-1),$$

– для топологии гиперкуб:

$$T^G_{p-p} = t_s + V \cdot t_w \cdot log_2\, p,$$
$$\grave{O}^G_{all-to-all} = \sum_{i=1}^{lp} (t_s + 2^{i-1} V t_w) = t_s \cdot log_2\, p + t_w \cdot V \cdot (p-1).$$

Анализ эффективности полученных параллельных алгоритмов производился на основе следующих показателей:

– времени решения при помощи последовательного алгоритма, T_1;

– времени решения при помощи параллельного алгоритма без учета - $T_{p,comp}$ и с учетом обменных операций - $T_p = T_{p,comp} + T_{p,comm}$, так называемые потенциальный и реальный параллелизм;

– коммуникационной сложности алгоритма в зависимости от выбранной топологии соединения процессоров и модели передачи данных;

– максимальной степени параллелизма, Dot;

– коэффициентов потенциального и реального ускорения, S и эффективности, E параллельного алгоритма, а также масштабируемости на основе функции изоэффективности, f_E.

Динамические характеристики параллельных алгоритмов ЯМРК существенно зависят от многих факторов: размера задачи (m) и времени обращения к правой части исходной СОДУ (T_f), типа параллельного компьютера, размерности процессорных полей (p), времени выполнения арифметических операций (t_{op}), временных коммуникационных констант (латентность, t_s и время передачи одного слова, t_w), топологии межпроцессорного соединения. Определим, что при подсчете динамических характеристик алгоритмов предполагалось, что $t_{ad} = t_{mul} = t_{op}$ – любая арифметическая операция с плавающей точкой выполняется за одно и тоже время независимо от вида операции, это предположение справедливо для большинства современных компьютеров RISC архитектуры. Под масштабируемостью понимается – возможность алгоритма обеспечить постоянное значение эффективности вычислений при увеличении числа процессоров (возможно за счет увеличения размерности задачи). Оценкой масштабируемости является функция изоэффективности[3], определяющая зависимость числа используемых процессоров для обеспечения постоянного уровня эффективности параллельных вычислений:

$$W = K \cdot f_E(p),$$

где W - есть показатель вычислительной сложности задачи (количество операций), K - есть коэффициент, зависящий только от значения показателя эффективности.

Построение функции изоэффективности – это попытка соединить в едином аналитическом выражении все

перечисленные характеристики и оценить степень их влияния на качество параллельного алгоритма. Более того, это инструмент сравнения различных параллельных алгоритмов решения одной и той же задачи и определения условий, где использование одного алгоритма становится предпочтительнее, чем других.

Анализ теоретического выполнения параллельных методов решения нелинейной задачи Коши на базе ЯМРК и проведенный численный эксперимент позволяют сделать выводы:

– наиболее эффективными параллельными методами с точки зрения эффективности и ускорения являются вложенные методы;

– преимущества методов на базе локальной экстраполяции проявляются при получении высокоточных решений ($10^{-15} - 10^{-20}$) и для СОДУ со сложными правыми частями.

Лучшие характеристики параллелизма близкие к потенциальным, практически линейное ускорение и единичная эффективность достигаются при сочетании нескольких факторов: сложной правой части, причем $T_f >> t_{op}$, выполнении условия $p \le Dop = m$, использования топологии гиперкуб с синхронным или даже асинхронным выполнением обменов. Определение реальных характеристик параллелизма осуществлялось с помощью пакета *Mathematica®* (Wolfram Research Inc.), численный эксперимент проводился на базе тестов[4] для СОДУ с использованием библиотеки *MPI.*

Численный эксперимент и проведенный сравнительный анализ динамических характеристик параллельных алгоритмов решения задачи Коши показал, что достаточно большая по сравнению с явными методами вычислительная сложность неявных методов не является препятствием для использования их в высокопроизводительных мультипроцессорных системах. Однако требуется тщательный учет всех составляющих параллельной системы для того, чтобы такое решение было масштабируемым и эффективным. В частности практически линейное ускорение и близкая к единичной эффективность достигаются при использовании синхроннной передачи данных в топологии гиперкуб.

Литература:

1. Grand Challenges: High performance computing and communications. A report by the Committee on Physical, Mathematical and Engineering Sciences, NSF/CIZE, 1800 G. Street NW, Washington, DC 20550, 1991.
2. В.В. Воеводин, Вл. В. Воеводин. Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608 с.
3. Kumar V., Gupta A. Analyzing scalability of parallel algorithms and architectures.// Journal of Parallel and Distributed Computing.,22(3), 1994, p.379-391.
4. Арушунян О.Б., Залеткин С.Ф., Калиткин Н.Н. Тесты для вычислительного практикума по обыкновенным дифференциальным уравнениям.// Вычислительные методы и программирование,2002,т.3,с.11-19.