УДК 519.6

Анализ масштабируемости параллельных алгоритмов численного решения задачи Коши

Назарова И.А.
Донецкий национальный технический университет
nazarova@r5.dgtu.donetsk.ua

Abstract

Nazarova I.A. Scalability analysis of parallel algorithms for numerical decision Cauchy's problem. *A parallel algorithm quantity cannot be evaluated apart from the architecture it is implemented on. The paper is devoted application isoefficiency scalability metrics to numerical decision of Cauchy's problem on based embedded explicit one-step methods. Obtained algorithms are realized on parallel structures with ring, mesh and hypercube topologies. The estimations of the execution time, acceleration, efficiency and scalability parallel solution are defined.*

Введение

При разработке параллельных алгоритмов численного решения любых сложных динамических задач принципиальным моментом является анализ качества использования параллелизма. Эффективность реализации параллельных вычислений во многом зависит от адекватности отображения вычислительного алгоритма на параллельную архитектуру, согласования свойств алгоритма и особенностей суперкомпьютера. Поэтому выполнение параллельного алгоритма не может быть изучено в отрыве от параллельной вычислительной системы, на которой он реализован.

На сегодняшний день огромное количество работ [1-8] посвящено анализу существующих и введению новых динамических характеристик, определяющих качество параллельного алгоритма. Заметим, что широко используемые в теории и практике параллельных вычислений показатели качества, такие, как ускорение, S_p и эффективность, E_p не являются независимыми. Например, увеличение числа процессоров вычислительной системы в пределах, не превышающих максимальную степень параллелизма, часто приводит к повышению ускорения и, как правило, уменьшению эффективности. Этот и многочисленные факты свидетельствуют о том, что предлагаемые характеристики не являются исчерпывающими, то есть не предоставляют полной информации о качестве параллельного алгоритма. Поэтому необходимо введение новых [6-8], дополнительных характеристик для оценки эффективности параллельных вычислений.

Разработка параллельных приложений в качестве своей цели может иметь не столько уменьшение времени выполнения, сколько обеспечение возможности решения задач большей размерности. Способность параллельного алгоритма эффективно использовать процессоры при увеличении сложности расчетов является важной характеристикой параллельных вычислений и называется масштабируемостью. Параллельный алгоритм является масштабируемым (*scalable*), если при росте числа процессоров он обеспечивает увеличение ускорения при сохранении постоянного уровня эффективности использования процессоров.

В данной статье исследование масштабируемости параллельных вычислений проиллюстрировано на примере решения задачи Коши на основе явных одношаговых схем с контролем погрешности на шаге. В качестве параллельной вычислительной системы использована многопроцессорная модель класса MPP с разными топологиями.

1. Изоэффективный анализ – мера масштабируемости параллельных алгоритмов

Существует несколько подходов для количественной оценки свойств масштабируемости параллельного алгоритма в совокупности с архитектурой, на которой он реализован. Наиболее применяемой является метрика, предложенная в [3-8] и основанная на введении функции равной эффективности или изоэффективности. Для этого определяется новая динамическая характеристика: T_o – накладные расходы на параллелизм (*total overhead*). Величина общих накладных расходов включает суммарные затраты всех процессоров параллельной системы, в том числе на последовательную часть распараллеленного алгоритма, реализацию обменов, непроизводительные расходы на синхронизацию и

время простоя из-за несбалансированности загрузки процессоров. Для данной параллельной системы T_o есть функция размерности задачи – m, числа процессоров – p, машинно-зависимых констант вычислений и обмена, и определяется, как:

$$T_o(m,p) = p \cdot T_p - T_1, \quad (1)$$

где T_1 – время, необходимое для решения задачи заданного размера на одном процессоре с помощью наилучшего последовательного алгоритма;

T_p – общее время реализации параллельного алгоритма на параллельной архитектуре:

$$T_p = T_{p,comp} + T_{p,comm}. \quad (2)$$

Используя введенное обозначение, можно получить новые выражения для времени параллельного решения задачи, ускорения и эффективности:

$$S_p = \frac{T_1}{T_p} = \frac{pT_1}{T_1 + T_o}, E_p = \frac{S_p}{p} = \frac{T_1}{T_1 + T_o} = \frac{1}{1 + T_o / T_1}. \quad (3)$$

Пусть $E_p = const$ задает необходимый уровень эффективности выполняемых вычислений, тогда:

$$T_o / T_1 = (1 - E) / E,$$

$$T_1 = E /(1 - E) \cdot T_0 = K(pT_p - T_1), \quad (4)$$

где $K = E /(1 - E)$ – есть коэффициент, зависящий только от значения показателя эффективности.

Порождаемую последним соотношением зависимость $m = f_E(p)$ между сложностью решаемой задачи и числом процессоров называют функцией изоэффективности (*isoefficiency function*). Выражение (4) – центральное соотношение изоэффективного анализа. Функция f_E определяет, как надо увеличивать размер задачи при увеличении числа процессоров для поддержания постоянной эффективности. Изоэффективная функция некоторых параллельных систем есть полином от размерности процессорного поля: $O(p^x)$, где $x \geq 1$. При $x = 1$ получаем линейный масштабируемый параллельный алгоритм.

Малая степень p в функции изоэффективности свидетельствует о высокой масштабируемости. Если же имеет место экспоненциальная (или любая другая быстро растущая функция) зависимость, то речь идет о слабо масштабируемых системах. Близким к линейно масштабируемому алгоритму считаются алгоритмы с функцией изоэффективности порядка $O(\log_x p)$, такие алгоритмы получили название квазилинейных.

Таким образом, построение функции изоэффективности – это попытка соединить в едином аналитическом выражении характеристики задачи, метода решения, параллельной архитектуры и оценить степень их влияния на качество параллельного алгоритма. В работе описанная метрика масштабируемости использована, как инструмент сравнения различных параллельных алгоритмов решения одной и той же задачи и определения условий, где применение одного алгоритма становится предпочтительнее, чем других.

При анализе эффективности параллельных вычислений необходимо учитывать временные машинно-зависимые параметры, влияющие на скорость выполнения параллельных алгоритмов. Такими параметрами являются времена выполнения: операции с плавающей точкой и операции межпроцессорного обмена. Количество операций с плавающей точкой в секунду (Floating Point Operations Per Second – *FLOPS*) является характеристикой производительности процессора. Величина *FLOPS* определяется экспериментально на основе различных тестов. Время выполнения операции с плавающей точкой будем обозначать:

$$t_{op} = 1 / FLOPS,$$

где *FLOPS* – производительность исследуемого процессора. При подсчете динамических характеристик алгоритмов считается, 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.

При исследованиях предполагалось, что параллельная вычислительная система – однородна, то есть состоит из p гомогенных процессоров и идентичных каналов связи. Для оценки времени выполнения операции передачи одного сообщения объемом *V* байт между двумя процессами, локализованными на различных процессорах при распределенной памяти, используется следующая линейная модель:

$$T_{p-p} = t_s + t_w \cdot V \cdot l, \; t_w = \frac{y}{B}, \quad (5)$$

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).

Программные приложения реализуют SPMD-стиль для мультикомпьютеров: одна программа – много данных. При проведении вычислительных экспериментов использован язык

программирования C++, и библиотека обмена сообщениями MPICH-1.2.5, реализация стандарта MPI для OS Windows. Коллективные операции обмена сообщениями для повышения производительности обменов реализованы для каждой из предложенных топологий отдельно.

2. Параллельное решение задачи Коши на основе явных одношаговых вложенных методов

Численно решается задача Коши для системы обыкновенных дифференциальных уравнений (СОДУ) первого порядка с известными начальными условиями:

$$\begin{cases} \dfrac{dy_1(x)}{dx} = f_1(x; y_1, y_2, ..., y_m),\ y_1(x_0) = y_{10}, \\ \dfrac{dy_2(x)}{dx} = f_2(x; y_1, y_2, ..., y_m), y_2(x_0) = y_{20}, \\ \quad ... \\ \dfrac{dy_m(x)}{dx} = f_m(x; y_1, y_2, ..., y_m), y_m(x_0) = y_{m0}. \end{cases} \quad (6)$$

Явный s-стадийный одношаговый метод Рунге-Кутты (ЯМРК) может быть описан следующей вычислительной схемой:

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h \cdot \sum_{i=1}^{s} b_i \bar{k}_i,\ \bar{k}_i = F(x_n + c_i h; \bar{g}_i), \\ \bar{g}_i = \bar{y}_n + h \cdot \sum_{j=1}^{i-1} a_{ij} \bar{k}_j,\ i = 1, ..., s. \end{cases} \quad (7)$$

Наиболее эффективным способом определения локальной апостериорной погрешности при решении задачи Коши для однопроцессорных машин являются методы вложенных форм, которые дополнительно требуют вычисления одного или нескольких стадийных коэффициентов при вычислении более точной аппроксимации решения на шаге.

Вложенные методы определяют две формулы Рунге-Кутты смежных порядков точности $r(\hat{r})$ и для интегрирования СОДУ имеют вид:

$$\begin{cases} \bar{y}_{n+1} = \bar{y}_n + h \cdot \sum_{i=1}^{s} b_i \cdot \bar{k}_i, \\ \hat{\bar{y}}_{n+1} = \bar{y}_n + h \cdot \sum_{i=1}^{s'} \hat{b}_i \cdot \bar{k}_i. \end{cases} \quad (8)$$

Для определенности полагаем, что число стадий метода более высокого порядка на единицу больше, чем число стадий оцениваемого метода: $s' > s$. Если основу вложенных форм составляют явные разностные схемы, то стадийные коэффициенты должны определяться по формулам:

$$\begin{cases} \bar{k}_i = F(x_n + c_i h; \bar{g}_i), \\ \bar{g}_i = \bar{y}_n + h \sum_{j=1}^{i-1} a_{ij} \cdot \bar{k}_j,\ i = \overline{1, s'}. \end{cases} \quad (9)$$

При разработке параллельного алгоритма вложенного явного метода Рунге-Кутты (ВЯМРК) использовалась иерархическая декомпозиционная методика и графы влияния [1].

Процесс вычислений по формулам (8)-(9) можно представить как решение двух подзадач: вычисление приближенного решения $\bar{y}_{n+1}(h)$ порядка r в точке x_{n+1} с шагом h и $\hat{\bar{y}}_{n+1}(h)$ порядка $\hat{r}$ в точке x_{n+1} с тем же шагом, максимальная степень параллелизма совпадает с размерностью СОДУ и равна $Dop = m$.

В свою очередь, макрооперации следующего уровня можно представить, как решение s' подзадач вычисления стадийных коэффициентов и двух подзадач вычисления приближенного решения.

Время последовательного явного ВМРК $r(\hat{r})$ включает время вычисления стадийных коэффициентов, обеих аппроксимаций решения и составляет:

$$T_1 = (s+1)mT_F + [s^2 m + 6sm + 2s + 4m]t_{op},$$

где $T_F = \sum_{i=1}^{m} T_{f_i}$ - время вычислений правой части системы (1).

Время выполнения параллельного алгоритма ВМРК $r(\hat{r})$ включает время выполнения арифметических операций:

$$T_{p,comp} = m_1(s+1)T_F + (s^2 m_1 + 6sm_1 + 2s + 4m_1)t_{op}$$

и обменов:

$$T_{p,comm} = s' T_{all-to-all}(m_1, p), \text{ где } m_1 = \lceil m/p \rceil.$$

Оценки потенциального параллелизма ВМРК близки к оптимальным, то есть практически линейное ускорение и единичная эффективность. Определим время передачи данных для алгоритма ВМРК для различных топологий, объем передаваемых данных равен $m_1 = \lceil m/p \rceil$:

– кольцо:

$$T_{p,comm} = s' \cdot [(t_s + \lceil m/p \rceil \cdot t_w) \cdot (p-1)];$$

– решетка:

$$T_{p,comm} = s' \cdot [2t_s(\sqrt{p} - 1) + \lceil m/p \rceil \cdot t_w \cdot (p-1)];$$

– гиперкуб:

$$T_{p,comm} = s' \cdot [t_s \cdot \log_2 p + t_w \cdot \lceil m/p \rceil \cdot (p-1)].$$

3. Исследование масштабируемости параллельных вложенных методов решения задачи Коши на основе явных одношаговых схем

Постановка вопроса об оценке эффективности распараллеливания при увеличении числа процессоров с сохранением объема вычислений не всегда является целью исследований, более того, некоторыми авторами [3] считается спорной. Действительно, рост производительности системы в целом обусловлен сбалансированностью вычислительной работы и обменов на ее фоне. Невыполнение этого условия – одна из причин ухудшения качества параллелизма с увеличением размерности процессорного поля. Поэтому выводы о качестве параллельного алгоритма следует делать на основе достижения приемлемой эффективности при распараллеливании на число процессоров, фактически необходимое для решения задачи в заданные временные сроки. Другими словами, речь идет об оценке масштабируемости параллельной системы, состоящей из алгоритма в комбинации с архитектурой, на которой он реализован.

В данной статье исследуется масштабируемость параллельных алгоритмов вложенных методов, как наиболее эффективного способа оценки локальной апостериорной погрешности для явных одношаговых схем численного решения нелинейной нежесткой задачи Коши. Математическим аппаратом, позволяющим решить поставленную задачу, является теория изоэффективного анализа.

Для построения функции изоэффективности необходимо определить общие накладные расходы на параллельный алгоритм:

$$T_o = (s+1)p\cdot log_2 p\cdot t_s + m\cdot(s+1)(p-1)\cdot t_w + 2s(p-1)\cdot t_{op}.$$

Тогда, основное соотношение изоэффективного анализа для рассматриваемого алгоритма принимает вид:

$$T_1 = K\cdot T_o \quad \Rightarrow$$

$$T_1 = (s+1)\cdot mT_F + (s^2 m + 6sm + 2s + 4m)\cdot t_{op} =$$

$$= K\cdot T_o \quad \Rightarrow$$

$$T_1 = K[\, p\cdot log_2\, p\cdot(s+1)\cdot t_s + m\cdot(s+1)(p-1)\cdot t_w +$$

$$+ 2s\cdot(p-1)\cdot t_{op}\,]$$

Используя полученные соотношения, определим функцию изоэффективности, то есть найдем зависимость $m = f_E(p)$, на основе которой можно вычислить, как необходимо увеличивать размерность задачи при увеличении числа процессоров для поддержания постоянной эффективности. То есть, выразим размерность СОДУ, определяющую сложность исходной задачи, как функцию от следующих параметров: количество процессорных элементов, временные константы алгоритма и ВС, порядок и число стадий явного метода, коэффициент масштабируемости $m = f_E(p; T_F, t_{op}, t_s, t_w; s; K)$.

Напомним, что $K = E/1-E$.

$$m = \frac{K[\,plog_2 p\cdot(s+1)\cdot t_s + 2s\cdot(p-1)\cdot t_{op}] - 2s\cdot t_{op}}{(s+1)\cdot T_F + (s^2+6s+4)\cdot t_{op} - K(s+1)(p-1)\cdot t_w} \quad (8)$$

Проанализируем это соотношение с учетом характеристик алгоритма и параллельной вычислительной системы. Заметим, что из $Dop = p_{max} = m$, следует, что число процессоров всегда должно быть меньше либо равно размерности исходной СОДУ во избежание непроизводительных расходов (простоя оборудования). Вне зависимости от соотношения любых параметров, размерность СОДУ должна быть положительным числом. Исходя из (8) очевидно, что числитель дроби есть число положительное, так как в самом худшем случае при $p = s = 2$ это требование выполняется. Знаменатель (8) должен быть строго больше нуля, поэтому в качестве верхней границы для числа процессоров имеем следующее выражение:

$$p < \frac{(s+1)\cdot T_F + (s^2+6s+4)\cdot t_{op} + K(s+1)\cdot t_w}{K(s+1)\cdot t_w}. \quad (9)$$

Проанализируем соотношение (8) с выведенными ограничениями. По приведенной выше классификации рассмотренный алгоритм является квазилинейным относительно числа процессоров, то есть обладает высокой масштабируемостью.

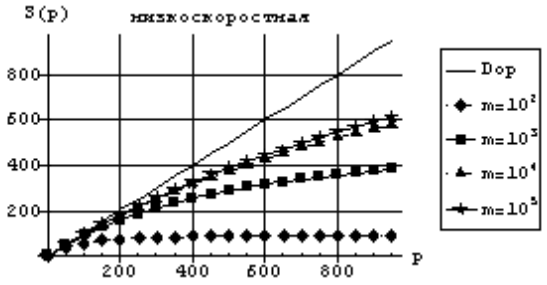

Рисунок 1 - Коэффициент ускорения ВЯМРК от числа процессоров для MIMD-систем с высокой латентностью коммутационной сети

Рисунок 1 представляет коэффициент ускорения при четырех различных значениях размерности задачи для асинхронных параллельных систем с низкоскоростными коммутационными сетями: высокая латентность и небольшая скорость передачи данных. Как и ожидалось, для малой размерности $m \le O(100\, t_{op})$ ускорение достигает пика уже при небольшом числе

процессоров: $p \sim 100$. За этой точкой увеличение размера процессорного поля существенного влияния на ускорение не оказывает. С другой стороны, ускорение для больших размерностей задачи близко к линейному.

Для MIMD систем с быстрыми каналами связи масштабируемость алгоритма повышается, то есть ускорение с ростом числа процессоров становится практически линейным, особенно для задач большой размерности (рис. 2). Пик ускорения достигается при гораздо большем числе процессоров, величина максимального возможного ускорения возрастает. Число процессоров, при котором происходит насыщение ускорения, прямо пропорционально сложности исходной задачи. Эта тенденция проявляется для параллельных систем любой архитектуры, эффективности коммуникационных сетей и топологии соединения процессоров в них.

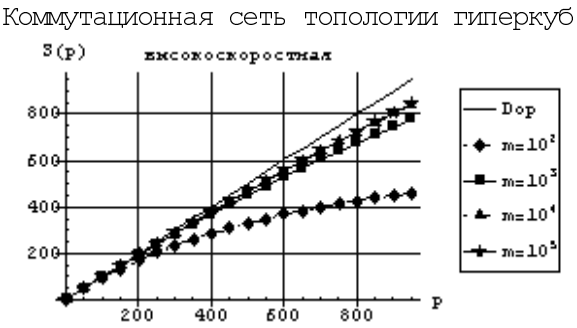

Рисунок 2 - Коэффициент ускорения ВЯМРК от числа процессоров для MIMD-систем с высокоскоростными коммутационными сетями

Рассмотрим влияние на величину функции изоэффективности нескольких параметров: коэффициента масштабируемости, а, следовательно, и приемлемой эффективности, порядка исследуемого метода, сложности правой части, при варьировании числа процессоров и характеристик коммутационной сети (рис. 3-6).

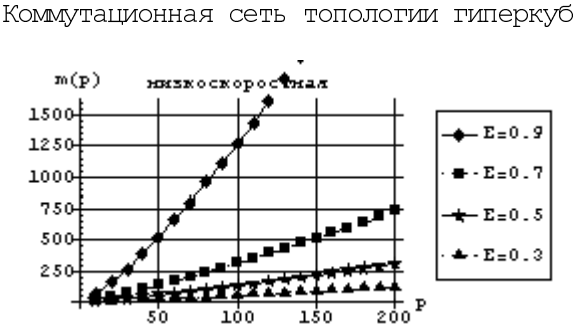

Рисунок 3 - Функция изоэффективности ВЯМРК для MIMD-систем с высокой латентностью при различных значениях коэффициента эффективности

Графики рисунков 3-4 демонстрируют, какой размерности задачу необходимо решать на имеющейся параллельной ВС, чтобы эффективность использования оборудования достигала определенного заданного значения. Диапазон изменения коэффициента эффективности $E = 0.3 \div 0.9$, позволяет исследовать загрузку процессоров системы от малой до практически полной.

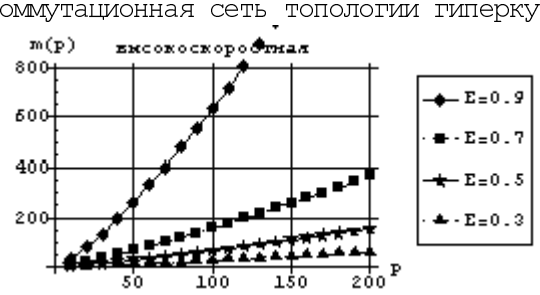

Рисунок 4 - Функция изоэффективности ВЯМРК для высокоскоростных MIMD-систем при различных значениях коэффициента эффективности

Так, при размерности процессорного поля $p = 100$ и низкоскоростных коммутационных сетях для достижения эффективности 0.9 нужна задача размерности не менее $m \approx 1250$, в то же время для $E = 0.7$ приблизительно в пять раз меньше $m \approx 223$. Достижение тех же характеристик для быстрых каналов передачи данных требует решения задач практически в несколько раз меньшей размерности.

На графиках рисунков 5-6 приведены зависимости функции изоэффективности от порядка наиболее известных вложенных методов и различных коммутационных сред, причем коэффициент масштабируемости равен $K = 9$, что соответствует $E = 0.9$.

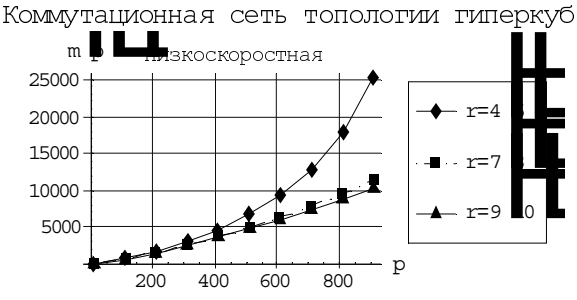

Рисунок 5 - Функция изоэффективности для MIMD-систем и ВЯМРК различных порядков точности при $E=0.9$ 1) Мерсона 4(5); 2) Фельберга 7(8); 3) Дормана-Принса 9(10)

Очевидно, что чем меньше порядок метода, тем большей размерности задачу необходимо решать для достижения одной и той

же эффективности при одинаковом числе используемых процессоров.

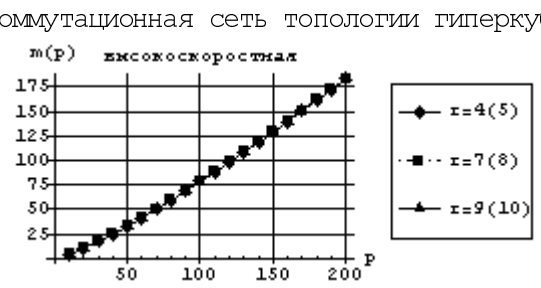

Рисунок 6 - Функция изоэффективности для MIMD-систем и ВЯМРК различных порядков точности при $E=0.9$ 1) Мерсона 4(5); 2) Фельберга 7(8); 3) Дормана-Принса 9(10)

Аналогичные тенденции поведения функции изоэффективности имеют место при изучении влияния на нее параметра исходных данных, а, именно, сложности правой части СОДУ. Все графики построены с учетом самых сильных ограничений на число процессоров, которые достигаются для низкоскоростных сетей.

Таким образом, применение аппарата изоэффективного анализа позволяет сделать выводы о качестве полученного параллельного алгоритма на основе единого аналитического выражения, функции изоэффективности, сократив тем самым многочисленные эксперименты при различных комбинациях параметров от которых зависят характеристики параллелизма.

Заключение

Предложенные и теоретически обоснованные в [8-9] параллельные методы оценки локальной апостериорной погрешности численного решения задачи Коши для систем обыкновенных дифференциальных уравнения на основе явных одношаговых разностных схем исследованы с точки зрения их масштабируемости.

На основе построенного аналитического выражения (функции изоэффективности), связывающего размерность СОДУ и размерность процессорного поля параллельной ВС, исследована масштабируемость полученных параллельных алгоритмов решения задачи Коши в зависимости от:

1) характеристик исходной задачи Коши, а именно, размерности, сложности правой части системы обыкновенных дифференциальных уравнений;

2) параметров используемой параллельной вычислительной системы:

- количества процессоров,
- типа параллельной ВС (SIMD, MIMD и кластерные системы),
- латентности и времени передачи данных в сетях различных топологий;

3) характеристик (порядка) используемого численного метода интегрирования СОДУ.

Полученные теоретические результаты и проведенный численный эксперимент свидетельствуют о высокой квазилинейной $O(\log_2 p)$, масштабируемости параллельного решения задачи Коши на основе ВЯМРК.

Перспективным направлением исследований является разработка, исследование эффективности и масштабируемости численного решения жестких СОДУ на основе неявних численных схем, а также проведение сравнительного анализа эффективности неявных одношаговых методов решения нелинейной задачи Коши на основе блочных k-точечных и традиционных (типа Рунге-Кутты) методов интегрирования жестких задач одного и того же порядка точности.

Литература

1. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608с.
2. Воеводин В.В. Математические проблемы параллельных вычислений // Тезисы докладов II Всероссийской научной конференции "Методы и средства обработки информации". 5-7 октября 2005. – Москва: МГУ, 2005. – С. 22-33.
3. Гергель В.П., Стронгин Р.Г. Основы параллельных вычислений для многопроцессорных вычислительных систем.– Н.Новгород: ННГУ, 2001.– 122с.
4. Гергель В.П. Теория и практика параллельных вычислений. – Москва: Бином. Лаборатория знаний, 2007. – 423с.
5. Grama A., Gupta A., Kumar V. Isoefficiency: Measuring the scalability of parallel algorithms and architectures // IEEE Parallel and Distributed technology, 2003. – P. 12-21.
6. Kumar V., Gupta A. Analyzing scalability of parallel algorithms and architectures // Journal of Parallel and Distributed Computing, 22(3), 1994. – P. 379-391(2nd edn., 2003).
7. Gupta A., Kumar V. Scalability of parallel algorithm for matrix multiplication // Technical report TR-91-54, Department of CSU of Minneapolis, 2004.
8. Фельдман Л.П., Назарова И.А. Параллельные алгоритмы численного решения задачи Коши для систем обыкновенных дифференциальных уравнений // Математическое моделирование, том 18, №6, 2006. - С. 17-31.
9. Фельдман Л.П., Назарова И.А. Эффективность параллельных алгоритмов оценки локальной апостериорной погрешности для численного решения задачи Коши // Электронное моделирование, т. 29, № 3, 2007. – С. 11-25.

Поступила в редколлегию 12.03.2009