Новые подходы к построению высокоэффективных параллельных алгоритмов для численного решения краевых задач на структурированных сетках*

В.М. Волохов, С.И. Мартыненко, П.Д. Токталиев, Л.С. Яновский

Институт Проблем Химической Физики РАН

Рассмотрены новые подходы к построению высокоэффективных параллельных алгоритмов для численного решения краевых задач. В качестве базового алгоритма выбрана Универсальная Многосеточная Технология – односеточный вариант метода Зейделя, позволяющий решать широкий класс прикладных задач с вычислительными усилиями близкими к оптимальным. Исследованы два подхода к распараллеливанию вычислений, основанных на комбинированном и чисто геометрическом построении предобуславливателя. Показаны преимущества данных подходов по сравнению с традиционными методами построения параллельных алгоритмов и получены оценки эффективности параллелизма.

Ключевые слова: параллельные вычисления, краевые задачи, многосеточные методы.

1. Введение

Основные трудности построения высокоэффективных параллельных алгоритмов для численного решения краевых и начально-краевых задач покажем на примере одномерной краевой задачи Дирихле для уравнения Пуассона на интервале (0,1):

$$u'' = -f(x), u(0) = a, u(1) = b$$

Построим на интервале (0,1) равномерную сетку, посредством разбиения отрезка [0,1] на N частей. Узлы сетки заданы соотношением

$$x_i = (i-1)h$$

где $h = 1/N$ есть шаг сетки. Построенная вычислительная сетка при $N = 8$ показана на рис. 1.

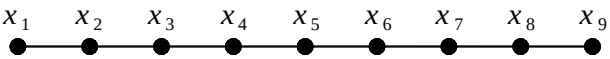

Рис. 1. Равномерная сетка ($N = 8$) для аппроксимации задачи Дирихле для уравнения Пуассона.

Пусть w_i есть сеточный аналог непрерывной функции $u(x)$, причем $w_i = u(x_i)$. Численное решение задачи Дирихле состоит в отыскании значений сеточной функции w_i во внутренних узлах сетки, т.е. при $i = 2,3, \dots, N$. Значения функции w_i на границе отрезка [0,1] являются известными: $w_1 = a$ и $w_{N+1} = b$.

Стандартная конечно-разностная аппроксимация второй производной на трехточечном шаблоне приводит к следующей разностной краевой задаче

$$\frac{w_{i-1}-2w_i+w_{i+1}}{h^2} = -f(x_i), i = 2,3, \dots, N, \quad (1)$$
$$w_1 = a, w_{N+1} = b$$

Фактически численное решение краевых задач сводится к решению плохообусловленных систем линейных алгебраических коэффициентов (СЛАУ) с разреженной матрицей коэффици-

* Исследовательские работы проводятся при финансовой поддержке государства в лице РНФ по соглашению №15-11-30012 от 08.07.2015 по теме: «Суперкомпьютерное моделирование физико-химических процессов в высокоскоростном прямоточном воздушно-реактивном двигателе гиперзвукового летательного аппарата на твёрдых топливах».

ентов высокого порядка [1]. Именно решение подобных СЛАУ требует наибольших вычислительных усилий по сравнению с остальными этапами вычислительного эксперимента.

Простейший способ построения параллельных алгоритмов основан на блочных предобуславливателях. Перепишем задачу (1) в матричной форме

$Ag = b,$

полагая, что граничные условия включены в матрицу коэффициентов A, а g есть вектор неизвестных. Если матрица коэффициентов A имеет блочную структуру, то сформируем матрицу W, диагональные блоки которой образованы диагональными блокам матрицы A. Тогда итерационный метод решения СЛАУ $Ag = b$ принимает вид

$$W\left(g^{(n+1)} - g^{(n)}\right) = b - Ag^{(n)}. \quad (2)$$

Если матрица W состоит из p блоков, то решение (2) сводится к решению p независимых СЛАУ меньшей размерности, которое может быть получено при помощи любого итерационного метода.

Положим, что используемая многопроцессорная система состоит из достаточного количества отдельных вычислительных устройств (процессоров), соединенных между собой некоторой коммутационной сетью. Далее не будут рассматриваться существующие или перспективные архитектуры многопроцессорных систем и способы обменов данными между отдельными процессорами, а также задержки связанные с коммуникационной сетью. Основной целью данной работы является формулировка новых подходов к построению высокоэффективных параллельных алгоритмов для численного решения краевых задач на структурированных сетках. Для решения данной задачи необходимо переформулировать задачу об итерационном решении СЛАУ таким образом, чтобы выделить достаточно большое количество независимых подзадач, которые можно решать параллельно на многопроцессорной системе с минимальными обменами данных.

Реализация параллельного алгоритма для решения СЛАУ (2) с блочно-диагональной матрицей W на p процессорах осуществляется следующим образом: сначала выполняется одна итерация (2), затем выполняется обмен данными между p процессорами.

В общем случае матрицу W можно построить алгебраическими методами (т.е. без привлечения информации о вычислительной сетке), но применительно к задаче (1) построение матрицы W означает разделение сетки на подсетки с «перехлестом». На рис. 2 показана последовательность действий в случае $p = 2$.

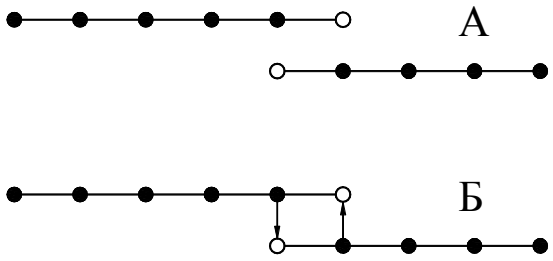

Рис. 2. Распараллеливание итераций (2): А) параллельное решение двух независимых СЛАУ меньшей размерности, Б) обмен данными.

Если исходная СЛАУ $Ag = b$ состоит из M уравнений, то количество блоков на диагонали матрицу W лежит в пределах от 2 до M и итерационный метод (2) может быть реализован на $p \in [2, M]$ процессорах. Случай $p = M$ соответствует методу Якоби с точечным упорядочением неизвестных. Применительно к разностной краевой задаче (1) итерации метода Якоби принимают вид

$$w_i^{(n+1)} = \frac{1}{2}\left(w_{i-1}^{(n)} + w_{i+1}^{(n)} + f(x_i)h^2\right), \quad (3)$$

где n есть номер итерации метода Якоби. Если задано некоторое начальное приближение к решению при $n = 0$, то, согласно (3), все значения $w_i^{(n+1)}$ можно вычислить независимо друг от друга, поэтому метод Якоби иногда рассматривают в качестве прототипа параллельного алгоритма [2].

Следуя [2], введем следующие меры параллелизма:

Определение 1. Ускорением параллельного алгоритма называется отношение времени выполнения алгоритма на одном процессоре $T(1)$ к времени выполнения алгоритма в системе из $\mathbf{p}$ процессоров $T(p)$

$$S_p = \frac{T(1)}{T(p)}$$

Определение 2. Эффективностью параллельного алгоритма называется величина

$$E_p = \frac{S_p}{p}.$$

На первый взгляд может показаться, что для метода Якоби ожидается $S_p \to p$ и $E_p \to 1$ (полный параллелизм), однако в действительности этого не происходит. Если используется достаточно большое количество процессоров $p \to M$, то возрастают издержки, связанные с обменами данных. Кроме того, скорость сходимости последовательного метода Якоби невелика. Как следует из (3) скорость распространения информации о граничных условиях внутрь области составит шаг сетки за одну итерацию при точечном упорядочении неизвестных, т.е. количество итераций метода Якоби, необходимых для решения разностной краевой задачи (1), должно превышать $N/2$ (рис. 3). Поскольку для выполнения одной итерации необходимо выполнить $O(N)$ арифметических операций, то общий объем вычислительной работы, необходимой для получения численного решения, будет более $O(N^2)$ арифметических операций. Поэтому даже в параллельном исполнении метод Якоби требует чересчур большого объема вычислений из-за медленной скорости сходимости последовательного алгоритма. Метод Якоби часто применяют в параллельных алгоритмах для решения начально-краевых задач, если для их аппроксимации использована явная разностная схема. Данный эффект, связанный с ухудшением качества предобуславливания при увеличении количества процессоров, характерен и для других итерационных методов.

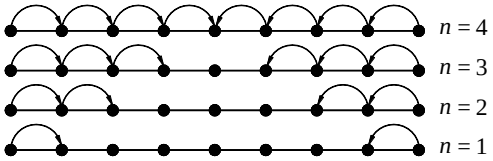

Рис. 3. Распространение информации о граничных условиях при решении разностной краевой задачи (1) методом Якоби (3) с точечным упорядочением неизвестных в зависимости от итерации n.

Чем сложнее последовательный алгоритм по своей конструкции, тем сложнее построить его эффективный параллельный вариант. Поэтому настоящим прорывом можно назвать многосеточный метод Р.П. Федоренко, в котором используется особенность сходимости простейших итерационных методов (типа Якоби и Зейделя) для достижения оптимальной (или неулучшаемой) скорости сходимости последовательного алгоритма [3]. Р.П. Федоренко показал, что скорость сходимости отдельных итерационных методов не зависит от шага сетки h на первых итерациях. Итерационные методы, обладающие подобным свойством, получили название сглаживатели. Поэтому если решать разностную краевую задачу на нескольких сетках, причем на каждой выполнять несколько сглаживающих итераций, то возможно достичь оптимальной скорости сходимости последовательного алгоритма, т.е. решить разностную краевую задачу выполнив всего $O(N)$ арифметических операций, где N есть количество узлов сетки. На рис. 4 показана самая мелкая сетка G^0 и грубые сетки G^1 и G^2 построенные удвоением шага. Верхний индекс указывает на принадлежность к сетке. Многосеточные методы подробно описаны в [4].

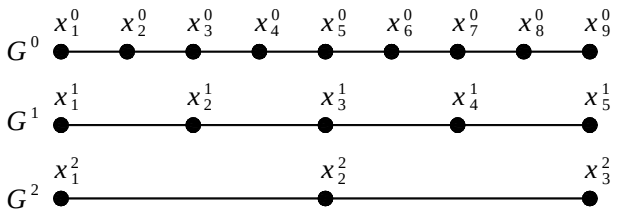

Рис. 4. Мелкая G^0 и грубые (G^1и G^2) сетки многосеточного метода.

Многосеточные методы сразу позволили сформулировать дополнительные меры параллелизма алгоритмов, предназначенных для решения прикладных задач [2]:

Определение 3. Ускорением параллельного алгоритма по сравнению с наилучшим последовательным алгоритмом называется отношение

$$\tilde{S}_p = \frac{\tilde{T}(1)}{T(p)},$$

где $\tilde{T}(1)$ есть время выполнения быстрейшего последовательного алгоритма на одном процессоре и $T(p)$ есть время выполнения параллельного алгоритма на системе из p процессоров.

Определение 4. Эффективностью параллельного алгоритма по отношению к наилучшему последовательному алгоритму называется величина

$$\tilde{E}_p = \frac{\tilde{S}_p}{p},$$

Определения 1 и 2 характеризуют ускорение и эффективность параллельного алгоритма и представляют теоретический интерес, в то время как определения 3 и 4 позволяют оценить уменьшение времени счета параллельного алгоритма по сравнению с наилучшим последовательным, т.е. являются теми мерами параллелизма, которые представляют при решении практических задач. За $\tilde{T}(1)$ можно выбрать время выполнения наиболее распространенного варианта многосеточных методов V-цикла [4].

Однако и многосеточные методы не позволили построить эффективный параллельный алгоритм для численного решения краевых задач. Как следует из рис. 4, количество узлов на грубых сетках уменьшается со скоростью геометрической прогрессии, поэтому по мере перехода к более грубым сеткам возрастают издержки, связанные с обменом данными. Количество узлов на самых грубых сетках может быть меньше количества используемых процессоров, поэтому часть процессоров будет неизбежно простаивать при выполнении сглаживающих итераций на самых грубых сетках. Все это приводит к снижению эффективности параллельного многосеточного алгоритма.

Оптимальная скорость сходимости классических многосеточных методов обусловлена оптимальной адаптацией их компонент к решаемой задаче. Долгое время не удавалось построить универсальный (робастный) многосеточный алгоритм, предназначенный для решения широкого класса (не)линейных краевых задач. В 90-е годы такой алгоритм был разработан и получил название Универсальная Многосеточная Технология (УМТ) [5]. В основе УМТ лежит применение основного многосеточного принципа (об аппроксимации длинноволновых компонент ошибки на грубых сетках) в односеточных алгоритмах. Использование только одной сетки для отыскания поправки позволило минимизировать количество проблемно-зависимых компонент технологии, снизить требования к выбору сглаживающей процедуры и избежать простоя процессоров в параллельном исполнении. Как следствие скорость сходимости УМТ лишь близка к оптимальной, т.е. для решения разностных краевых задач необходимо выполнять $O(N\,lgN)$ арифметических операций, где N есть количество узлов сетки. Для наглядности и простоты описания УМТ используют традиционную терминологию, принятую в многосеточных методах, однако этого можно избежать, поскольку УМТ является односеточным алгоритмом [5].

Отличительной особенностью параллельной УМТ является построение матрицы W в (2) не только алгебраическими (на самой мелкой сетке), но и геометрическими (на грубых сетках) методами [6,7]. На грубых сетках удалось построить блочно-диагональный предобуславливатель W без «перехлеста» сетки, т.е. достичь почти полного параллелизма.

Целью данной статьи является описание, теоретическое обоснование и экспериментальная иллюстрация преимуществ параллельной УМТ по сравнению с другими методами построения параллельных алгоритмов для численного решения краевых задач на структурированных сетках.

2. Основные компоненты УМТ

В УМТ самая мелкая сетка G_1^0 состоит из двух множеств точек $G^{\mathrm{v}}(0;1)$ и $G^{\mathrm{f}}(0;1)$, которые, если сетка G_1^0 является равномерной, заданы соотношениями:

$$G^{\mathrm{v}}(0;1) = \left\{x_i^{\mathrm{v}} | x_i^{\mathrm{v}} = (i-1)h, i = 1,2,\dots,N^0+1, h = 1/N^0\right\},$$
$$G^{\mathrm{f}}(0;1) = \left\{x_i^{\mathrm{f}} | x_i^{\mathrm{f}} = \frac{x_i^{\mathrm{v}} + x_{i+1}^{\mathrm{v}}}{2}, i = 1,2,\dots,N^0\right\}.$$

Точки и могут быть как узлами сетки, так и гранями контрольных объемов при аппроксимации краевой задачи интегро-интерполяционным методом [8]. Сетка G_1^0, построенная при $N^0 = 8$, показана на рис. 5.

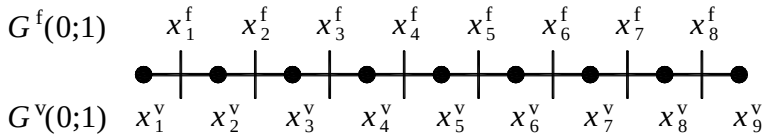

Рис. 5. Вычислительная сетка для аппроксимации интегро-интерполяционным методом.

Грубые сетки в УМТ строят не удвоением шага, а утроением, как показано на рис. 6. При этом грубые сетки G_1^1, G_2^1 и G_3^1 не имеют общих узлов, поэтому сглаживающие итерации можно проводить параллельно вне зависимости от используемого итерационного метода. Самая мелкая сетка G_1^0 образует нулевой сеточный уровень, а три грубые сетки G_1^1, G_2^1 и G_3^1 образуют первый сеточный уровень. Фактически при помощи грубых сеток G_1^1, G_2^1 и G_3^1 можно построить блочный предобуславливатель W в (2), но при этом отсутствует «перехлест» сетки, замедляющий скорость сходимости итерационных методов. Если быть точным, то в УМТ все вычисления проводят на исходной сетке, а термин «сеточный уровень» определяет лишь упорядочение неизвестных и величину погрешности аппроксимации [9].

Процесс построения еще более грубых сеток осуществляется рекуррентным образом, полученная таким образом иерархия сеток получила название многосеточной структуры (рис. 7). Самые грубые сетки состоят из нескольких узлов и образуют уровень L^+.

Аппроксимация краевых задач на многосеточной структуре осуществляется при помощи интегро-интерполяционного метода, результирующая СЛАУ имеет вид (3) с блочно-диагональным предобуславливателем W. Количество блоков на главной диагонали матрицы W составляет 3^{dl}, где d есть размерность задачи, а $l = 0,1,\dots,L^+$ есть номер сеточного уровня. Поэтому количество процессоров для параллельной УМТ должно составить $p = 3^{dl}$, где $l^* = 1,2,\dots,L^+$ есть номер сеточного уровня, начиная с которого источником параллелизма является многосеточная структура. В дальнейшем параметр l^* будет называться глубиной распараллеливания [9].

3. Параллельная УМТ

Рассмотрим параллельную УМТ с $l^* = 1$ (первая глубина), т.е. в трехмерном случае $d = 3$ необходимо $p = 27$ процессоров. На уровнях $l > l^*$ источником параллелизма является многосеточная структура (т.е. предобуславливающую матрицу строят геометрическим методом), а на уровнях $l < l^*$ (в данном случае это самая мелкая сетка) предобуславливающую матрицу строят алгебраическим методом.

Распределение сеток первого уровня и их подсеток среди процессоров и схема распараллеливания вычислений при $l \geq l^* = 1$ показана на рис. 8 и 9. В этом случае исходная задача естественным образом распадается на 27 независимых задач, почти одинакового размера, которые могут быть решены параллельно без обмена данными. Почти одинаковый размер задач позволяет добиться почти равномерной загрузки процессоров. Более того, на сетках уровней $l \geq l^* = 1$ возможно проведение дополнительных q^* многосеточных итераций, который позволят еще уменьшить объем пересылаемых данных по сравнению с объемом вычислительной работы.

При выполнении сглаживающих итераций на самой мелкой сетке ($l = 0$) необходимо искусственно сгруппировать неизвестные в 27 блоков для построения блочно-диагонального предобуславливателя W. Другими словами, на самой мелкой сетке матрицу W можно построить алгебраическими методами. Данное обстоятельство приводит к необходимости использования следующих мер параллелизма:

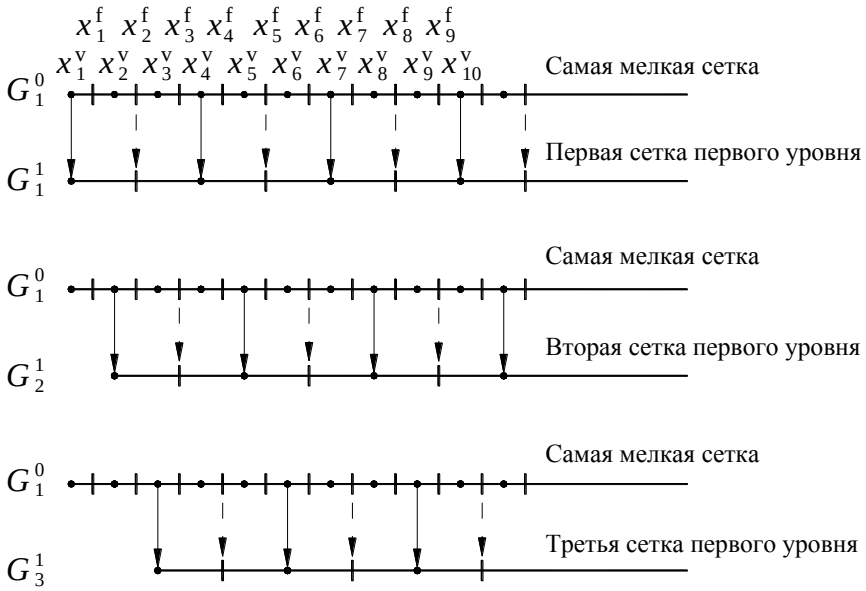

Рис. 6. Построение грубых сеток в УМТ.

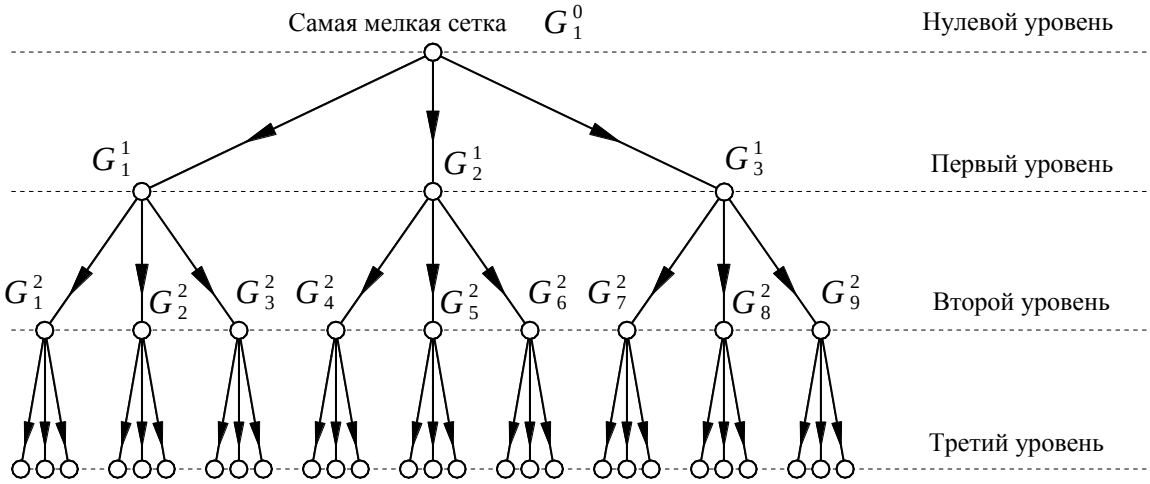

Рис. 7. Многосеточная структура.

Определение 5. Ускорением параллельного сглаживания на сетках уровня l по отношению к последовательному сглаживанию на тех же сетках называется величина

$$\bar{S}_l = \frac{T_l(1)}{T_l(p)},$$

где $T_l(1)$ есть время выполнения последовательного сглаживания (и оператора перехода) на одном процессоре, а $T_l(p)$ – время выполнения параллельного сглаживания (и оператора перехода) на системе из p процессоров.

Определение 6. Эффективностью параллельного сглаживания на сетках уровня l по отношению к последовательному сглаживанию на тех же сетках называется величина

$$\bar{E}_l = \frac{\bar{S}_l}{p}.$$

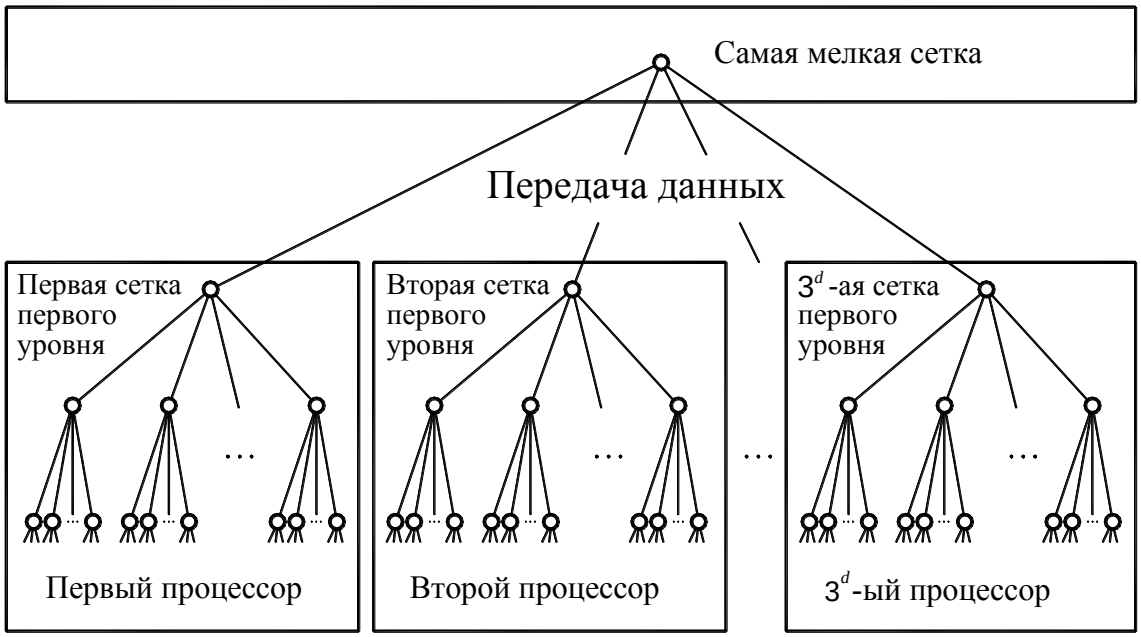

Рис. 8. Распределение грубых сеток среди процессоров в случае $l^* = 1$

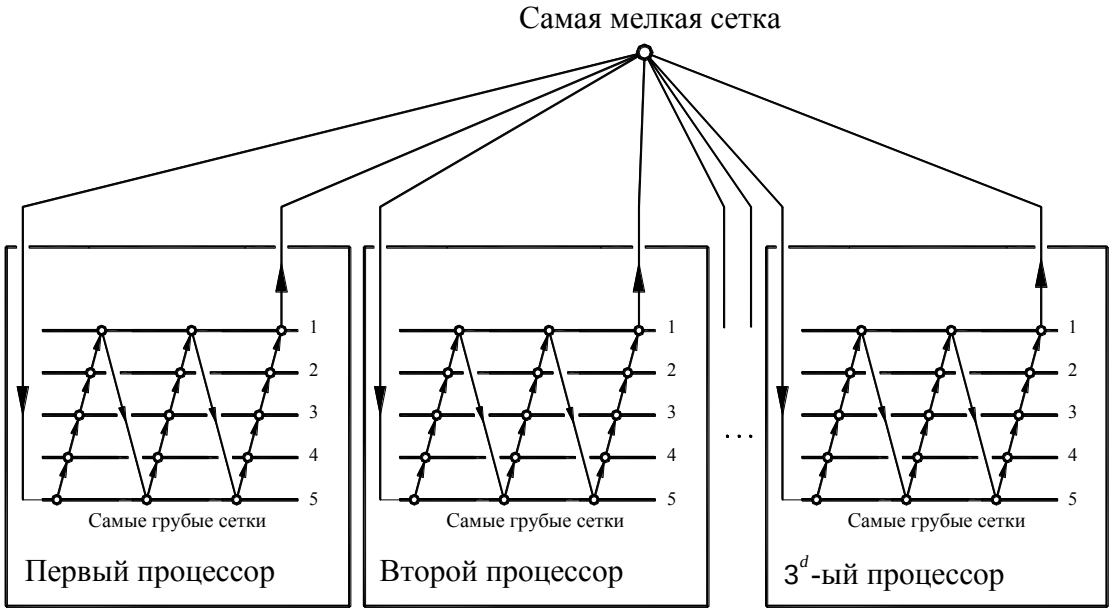

Рис. 9. Параллельное сглаживание на грубых сетках в случае $l^* = 1, q^* = 2$

Теоретическая оценка ускорения и эффективности параллельной УМТ может быть получена при допущении, что на всех уровнях выполнено одинаковое количество сглаживающих итераций, а на грубых сетках ($l > l^*$) УМТ обладает полным параллелизмом. Из определений 1 и 2 непосредственно следует, что для параллелизма первой глубины ($l^* = 1$) справедливо

$$S_p = 3^d E_p < 3^d \frac{1+q^* L^+}{\frac{1}{\bar{E}_0}+q^* L^+}. \quad (4)$$

Из полученной оценки следует, что эффективность параллельной УМТ зависит исключительно от эффективности параллельных сглаживающих итераций на самой мелкой сетке ($\bar{E}_0$), т.е. от совершенства предобуславливателя W, построенного алгебраическими методами. При этом итоговая эффективность параллельной УМТ будет больше, чем эффективность параллельных сглаживающих итераций на самой мелкой сетке.

$$E_p < E_{max} = \bar{E}_0 \frac{1 + q^* L^+}{1 + q^* L^+ \bar{E}_0} \Rightarrow \frac{E_{max}}{\bar{E}_0} = \frac{1 + q^* L^+}{1 + q^* L^+ \bar{E}_0} > 1 \Rightarrow E_{max} > \bar{E}_0.$$

Кроме того, увеличение количества узлов сетки приводит к повышению эффективности параллельной УМТ: $L^+ \to \infty \Rightarrow E_p \to 1$.

Несколько иначе выглядят характеристики УМТ при использовании мер параллелизма, данных в определениях 3 и 4. Сравнение времени счета с последовательным V-циклом приводит к следующей оценке

$$\tilde{S}_p = 3^d \tilde{E}_p < \frac{3^d}{1 - 2^{-d}} \frac{1}{\frac{1}{\bar{E}_0} + q^* L^+}. \quad (5)$$

Поскольку $L^+ = O(\lg N)$, то при $l^* = 1 \Rightarrow p = 3^d$ и $L^+ \to \infty \Rightarrow \tilde{E}_p = (lg^{-1} N)$, т.е. проигрыш во времени счета последовательному V-циклу будет возрастать с увеличением количества узлов.

Для вычислительного эксперимента воспользуемся первой краевой задачей для трехмерного уравнения Пуассона в единичном кубе, которая имеет точное решение $exp(x + y + z)$. Вычислительная сетка состоит из 245^3 = 14 706 125 узлов (L^+=3). Распараллеливание многосеточных итераций УМТ осуществлено при помощи технологии OpenMP с привлечением 27 ядер. На каждой сетке выполнены четыре сглаживающие итерации, на сетках уровней $l > 0$ выполнены две многосеточные итерации $q^* = 2$. Согласно результатам вычислительного эксперимента эффективность распараллеливания сглаживающей процедуры на самой мелкой сетке и на сетках уровней $l > 0$ составила $\bar{E}_0 = 0.89$ и $\bar{E}_* = 0.95$ соответственно. Эффективность параллельной УМТ составила 0.92, в то же время, согласно оценке (4), получим $S_p = 26.6$ и $E_p = 0.98$. С другой стороны, если сравнивать с последовательным V-циклом при одинаковом количестве многосеточных итераций, то эффективность параллельной УМТ составит 0.112, в то время как согласно оценке (5) $\tilde{E}_p = 0.125$. Таким образом, время решения модельной краевой задачи для уравнения Пуассона при помощи параллельной УМТ с привлечением 27 процессоров оказывается в три раза меньше, чем при помощи последовательного V-цикла.

Таким образом, построение предобуславливателя W геометрическими методами на грубых сетках и алгебраическими на мелких позволяет построить достаточно эффективный параллельный алгоритм для численного решения задач на структурированных сетках. Распараллеливание вычислений на грубых сетках не зависит от выбора сглаживающей процедуры.

4. Вычислительный эксперимент

Очевидные преимущества построения предобуславливателя W геометрическими методами можно использовать в другом подходе к построению параллельной УМТ, основанном на редукции к независимым задачам. Положим, что краевая задача (1) решена на грубых сетках G_1^1, G_2^1 и G_3^1 и разностные решения u_i^h сходится к решению u задачи (1) со вторым порядком, т.е.

$\|u - u_i^h\| \le C(3h)^2 = 9Ch^2$.

Таким образом, решение вместо одной разностной задачи на сетке G_1^0 трех независимых задач на грубых сетках G_1^1, G_2^1 и G_3^1 приводит к потере одной значащей цифры. Нетрудно видеть, что решение девяти независимых задач на сетках второго уровня приведет к потере еще одной значащей цифры т.д. Таким образом, возможна редукция исходной разностной задачи к совокупности 3^{md}, $m \ge 1, d = 2,3$ независимых задач меньшего размера. С одной стороны, эта редукция позволяет практически полностью распаллеливать вычисления, причем независимо от решаемых задач. С другой стороны, появляется возможность последовательно решать совокупность задач меньшего размера, даже если исходная задача не размещается в памяти компьютера. Потерю точности при редукции исходной разностной задачи можно компенсировать применением более точных аппроксимаций на грубых сетках.

В качестве примера редукции исходной разностной задачи к совокупности независимых задач меньшей размерности, рассмотрим уравнение Пуассона

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} = -f(x, y, z)$$

в единичном кубе. Положим, что точное решение имеет вид

$$u(x,y,z) = \sin(2\pi kx)\sin(2\pi ky)\sin(2\pi kz),$$

которое однозначно определяет правую часть $f(x,y,z)$ и однородные граничные условия Дирихле. Параметр k предназначен для иллюстрации влияния погрешности аппроксимации производных на точность численного решения. Поведение функции $sin(2\pi kx)$ при различных значениях k показаны на рис. 10.

Для численного решения первой краевой задачи для уравнения Пуассона построена равномерная сетка 451×451×451, которая порождает пятиуровневую многосеточную структуру. Для аппроксимации уравнения Пуассона использована схема шестого порядка аппроксимации, основанная на аппроксимациях Паде. Однако в вычислительном эксперименте вместо разностной краевой задачи на сетке 451×451×451 решены 729 разностных задач на сетках ≈ 50×50×50. Поскольку аналитическое решение краевой задачи известно, то погрешность численного решения определим как

$$\varepsilon = \max_{i,j,k} |u(x_i,y_j,z_k) - u^h_{i,j,k}|,$$

где $u(x_i,y_j,z_k)$ и $u^h_{i,j,k}$ есть точное и численное решение соответственно. В качестве сглаживателя использован метод Зейделя с блочным упорядочением неизвестных, на каждой сетке выполнено три сглаживающие итерации, всего выполнено двадцать многосеточных итераций. Коррекция решения проводилась после пяти многосеточных итераций. Вычисления производились на системе с общей памятью, включающей 4 процессора AMD Opteron 6176 и объемом RAM 128 Gb.

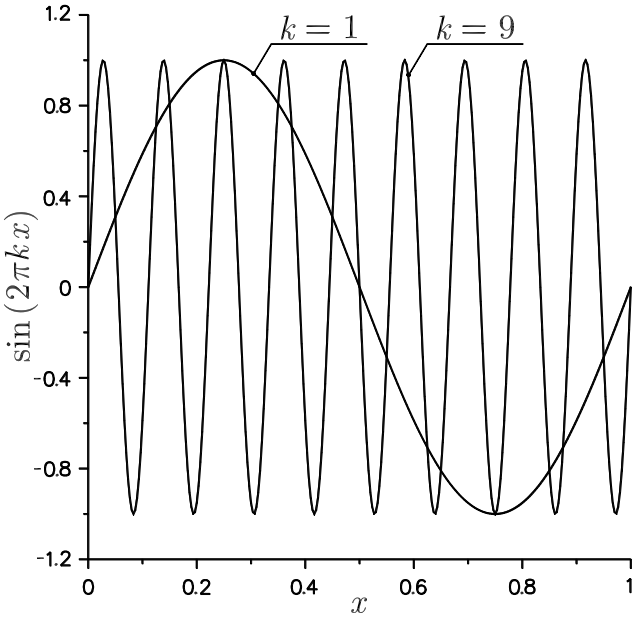

Рис. 10. Поведение функции $sin(2\pi kx)$ при различных значениях *k*.

Результаты вычислительного эксперимента показаны на рис. 11. При бóльших значениях параметра k наблюдается рост погрешности численного решения, связанный с увеличением абсолютных величин частных производных, входящих в главный член погрешности аппроксимации. Редукция приводит к уменьшению времени счета приблизительно в $(L^+ + 1)/(L^+ + 1 - k)$ раз, где k есть номер сеточного уровня с самыми мелкими сетками, используемый для отыскания поправки в многосеточных итерациях. Оценка минимальной эффективности параллелизма в данном случае основана на предположении, что время выполнения сглаживающих итераций пропорционально количеству узлов. Поскольку общее количество узлов сеток одного уровня есть $(N+1)^d$, $d = 2,3$, то время выполнения сглаживающих итераций в последовательном алгоритме есть

$$T(1) = a(N+1)^d,$$

где a есть коэффициент пропорциональности. Уровень l^* состоит из $3^{l^* d}$ сеток, имеющих разное количество узлов, что является причиной уменьшения эффектности параллелизма.

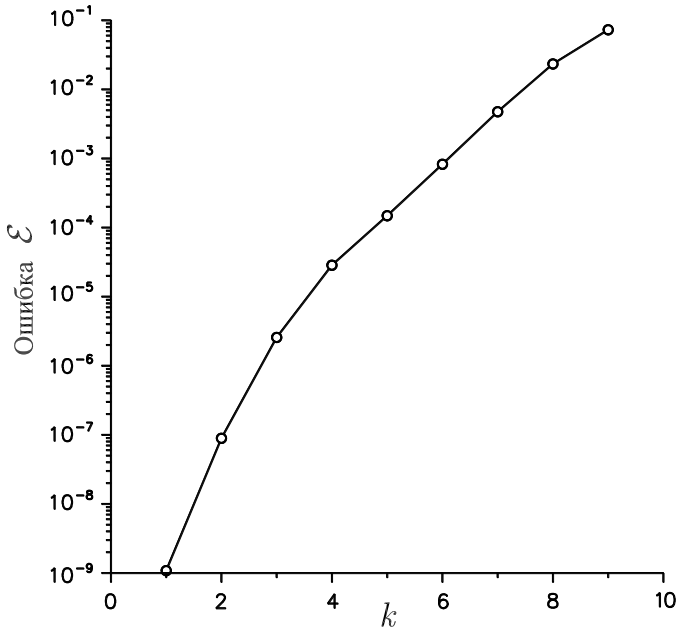

Рис. 11. Зависимость погрешности численного решения при различных значениях *k*.

Одна из сеток уровня l^* имеет максимальное количество узлов, которое можно оценить как

$$\approx \left(\left|3^{l^*}(N+1)\right|+1\right)^d < 3^{l^* d}(N+1+2\times 3^{l^*})^d.$$

Поэтому время выполнения $3^{l^* d}$ независимых задач на $p = 3^{l^* d}$ процессорах будет различным. Максимальное время выполнения сглаживающих итераций в параллельной УМТ ожидается на той сетке уровня l^*, которая состоит из наибольшего количества узлов, и составит

$$T_{\max}(p) = a3^{-l^* d}(N+1+2\times 3^{l^*})^d,$$

откуда нетрудно получить оценку минимальной эффективности параллелизма, основанного на редукции к $3^{l^* d}$ независимым задачам

$$E_{min} = \frac{1}{p}\frac{T(1)}{T_{max}(p)} = \left(1+2\frac{3^{l^*}}{N+1}\right)^{-d}.$$

В рассмотренном случае $l^* = 2, N = 450$, следует ожидать, что $E > E_{\min}$=90%.

5. Выводы

Если вычислительная сетка является структурированной, то возможно построить метод численного решения широкого класса прикладных задач, который будет высокоэффективен не только в последовательном, но и в многопроцессорном исполнении. Использование топологии вычислительной сетки позволяет строить предобуславливатель геометрическими методами, которые существенно превосходят алгебраические методы по возможности эффективного распараллеливания вычислений.

Литература

1. Самарский А.А., Гулин А.В. Численные методы. М.: Наука, 1989. 432 с.
2. Ортега Дж. Введение в параллельные и векторные методы решения линейных систем. М: Мир, 1991. 367 с.

3. Федоренко Р.П. Релаксационный метод решения разностных эллиптических уравнений // Вычислительная математика и математическая физика 1961. Т. 1, №5. С. 922–927.
4. Trottenberg U., Oosterlee C.W., Schüller A. Multigrid. L.: Academic Press, 2001.
5. Мартыненко С.И. Универсальная многосеточная технология для численного решения краевых задач на структурированных сетках // Вычислительные методы и программирование 2000. Т. 1, разд. 1. С. 85–104.
6. Мартыненко С.И. Распараллеливание универсальной многосеточной технологии // Вычислительные методы и программ. 2003. Т. 4, разд. 1. С. 45–51.
7. Мартыненко С.И. Оценки эффективности распараллеливания универсальной многосеточной технологии // Вестник МГТУ: Ест. науки. 2011. №4. С. 63–80.
8. Самарский А.А. Теория разностных схем: Учеб. пособие. М.: Наука. Физматлит, 1983. 616 с.
9. Мартыненко С.И. Многосеточная технология: теория и приложения. М.: ФИЗМАТЛИТ, 2015. 208 с.

New approaches for construction of highly efficient parallel algorithms for numerical solution of the boundary value problems on structured grids

V.M. Volokhov, S.I. Martynenko, P.D. Toktaliev, L.S Yanovskiy

The Institute of Problems of Chemical Physics of the Russian Academy of Sciences

New approaches for development of high efficient parallel algorithms for numerical solving boundary value problems have been considered. Robust Multigrid Technique (single grid variant of Seidel method for solving a large class of applied problems with close-to-optimal computational efforts) is taken as a basic algorithm. Two approaches for parallelisation of computations based on combined and purely geometric preconditioning have been studied. Advantages of these approaches over traditional methods of constructing parallel algorithms and estimations of parallelism efficiency are given.

Keywords: parallel computing, boundary value problems, multigrid methods.

References

1. Samarsky A.A., Gulin A.V. Numerical methods. M.: Nauka, 1989. 432 p. (in Russian)
2. Ortega J. Introduction to parallel and vector methods of solving linear systems. M: Mir, 1991. 367 p. (in Russian)
3. Fedorenko R.P. The speed of convergence of one iterative process // USSR. J. Comput. Math. and Math. Phys. 4 (1964), no.3, pp.227–235.
4. Trottenberg U., Oosterlee C.W., Schüller A. Multigrid. L:Academic Press, 2001.
5. Martynenko S.I. Robust multigrid technique for solving partial differential equations on structured grids // Numerical Methods and Programming, 2000. Vol. 1, sec. 1. pp. 85-104.
6. Martynenko S.I. Parallelization of the robust multigrid technique // Numerical Methods and Programming, 2003. Vol. 4, sec. 1. pp. 45-51. (in Russian)
7. Martynenko S. I. Estimations of parallezation efficiency of robust multigrid technique // Herald of the Baumann Moscow State Technical University, Natural Sciences, 2011. №. 4. pp. 63-80. (in Russian)
8. Samarskii A.A. Theory of difference schemes. M.: Nauka. Fizmatlit, 1983. 616 p. (in Russian)
9. Martynenko S.I. Multigrid technology: theory and applications. M.: FIZMATLIT, 2015. 208 p. (in Russian)