Исследование эффективности применения графических ускорителей при распараллеливании решения трехмерных краевых задач на квазиструктурированных сетках*

И.А. Климонов[2], В.Д. Корнеев[1], В.М. Свешников[1,2]

Институт вычислительной математики и математической геофизики СО РАН[1],
Новосибирский государственный университет[2]

При распараллеливании решения трехмерных краевых задач на квазиструктурированных сетках методом декомпозиции расчетной области на подобласти, сопрягаемые без наложения, наиболее трудоемкой вычислительной процедурой является решение краевых подзадач в подобластях. Использование параллелепипедальных квазиструктурированных сеток дает возможность применить для этих целей быстросходящиеся методы переменных направлений. Распараллеливание итерационного процесса по подобластям проводится на CPU в системе MPI, а для решения подзадач в настоящей работе предлагается использовать графические ускорители GPU. Проводятся экспериментальные исследования применения графических ускорителей при решении подзадач методом Писмана – Рэчфорда. Даются экспериментальные оценки ускорения распараллеливания в гибридной вычислительной среде CPU + GPU по сравнению с расчетами только на CPU.

Ключевые слова: краевые задачи, методы декомпозиции области, уравнение Пуанкаре-Стеклова, квазиструктурированные сетки, метод Писмана – Рэчфорда, графические ускорители.

1. Введение

Основой распараллеливания решения трехмерных краевых задач, рассматриваемого в настоящей работе, является метод декомпозиции расчетной области [1,2] на параллелепипедальные подобласти, сопрягаемые без наложения. В каждой такой подобласти строится своя равномерная параллелепипедальная подсетка. Совокупность подсеток образует квазиструктурированную сетку. Ее достоинством является, с одной стороны, простота использования и, с другой стороны, адаптивность к решению за счет регулировки плотности узлов подсеток. Сшивка решений в подобластях осуществляется путем прямой аппроксимации и решения уравнения Пуанкаре – Стеклова на границе сопряжения подобластей (интерфейсе), что приводит к итерационному процессу, на каждом шаге которого необходимо решать краевые подзадачи. Отметим, что для двумерной постановки данный подход рассматривался в работах [3 – 5].

Подсетки группируются в объединения, содержащие приблизительно одинаковое число узлов, для балансировки загрузки процессоров многопроцессорной суперЭВМ. Строится отображение: «одно объединение – один процессор», согласно которому вычислительный процесс распараллеливается на CPU в системе MPI [6]. Наибольшая вычислительная нагрузка при этом приходится на решение краевых подзадач. В настоящей работе предлагается и экспериментально исследуется способ ускорения данных вычислений, а именно применение для этих целей графических ускорителей GPU, то есть решение всей задачи по сути дела проводится в гибридной вычислительной среде CPU+GPU. В качестве метода решения подзадач был выбран трехмерный аналог метода Писмана – Рэчфорда [7], который легко распараллеливается, так как состоит из независимых прогонок по различным направлениям. В работе рассматриваются тех-

* Работа выполнена при финансовой поддержке Российского научного фонда (проект № 14-11-00485) и РФФИ (проект № 16-01-00168)

нологии его реализации в системе CUDA [8] и даются результаты численных экспериментов, показывающие значительное (более 60 раз) ускорение вычислений по сравнению с расчетами только на CPU.

2. Постановка задачи и основы алгоритма ее решения

Пусть в замкнутой трехмерной области $\overline{G} = G \bigcup \Gamma$ с границей Γ требуется решить краевую задачу

$$\Delta u = g_1, \quad l u\big|_{\Gamma} = g_2. \tag{1}$$

Здесь $u = u(T)$ – искомая функция, $g_1 = g_1(T)$, $g_2 = g_2(T)$ – заданные функции ($T = (x, y, z)$ – текущая точка, где x, y, z – декартовы координаты), Δ – оператор Лапласа, l – оператор граничных условий. Рассматриваются граничные условия Дирихле, Неймана, а также смешанные краевые условия. Предполагается, что граница Γ и функции g_1, g_2 таковы, что существует единственное решение задачи (1), обеспечивающее гладкость, достаточную для проведения дальнейших рассуждений.

Построим в расчетной области $\overline{G}$ структурированную равномерную макросетку $\overline{\Omega}_H$ с шагами, намного превышающими максимальный шаг результирующей сетки, на которой ищется решение исходной задачи. Тем самым мы проведем декомпозицию $\overline{G}$ на непересекающиеся подобласти $\overline{G}_m$, $m = \overline{1, M}$,где M – известное целое число. Граница сопряжения подобластей (интерфейс) γ, в свою очередь, разбивается на грани γ_f, ребра γ_e и макроузлы γ_m, являющиеся узлами макросетки $\overline{\Omega}_H$ так, что $\gamma = \gamma_f \bigcup \gamma_e \bigcup \gamma_m$. Для дальнейшего удобно ввести объединение $\gamma_{e,m} = \gamma_e \bigcup \gamma_m$ и область $G_0 = G \setminus \gamma$.

Построим в подобластях $\overline{G}_m$ структурированные равномерные подсетки $\overline{\Omega}_{h,m}$. Объединение этих подсеток составляет квазиструктурированную сетку

$$\overline{\Omega}_h = \bigcup_{m=1}^{M} \overline{\Omega}_{h,m}.$$

Исходную краевую задачу (1) переформулируем следующим образом: в замкнутой области $\overline{G}$ требуется найти решение уравнения Пуанкаре – Стеклова

$$Fv(T) = 0, \quad T \in \gamma_f, \tag{2}$$

совместно с решением краевых задач

$$\Delta u(T) = g_1(T), \quad T \in \gamma_{e,m} \tag{3}$$

$$\Delta u(T) = g_1(T), \quad l u\big|_{\Gamma} = g_2, \quad u\big|_{\gamma} = v, \quad T \in G_0 \tag{4}$$

относительно функций u и v. Здесь оператор F определяется как

$$Fv \equiv \left(\frac{\partial u(v)}{\partial \vec{n}} \right)_{\gamma_f}^{(+)} - \left(\frac{\partial u(v)}{\partial \vec{n}} \right)_{\gamma_f}^{(-)}, \tag{5}$$

где v – след функции u на γ (в том числе на γ_f).

Решение задачи (2) – (4) будем проводить методом итераций по подобластям, состоящим из следующих этапов.

1. Задается начальное приближение $v_f^{(0)}$ на гранях γ_f.
2. Из уравнения (3) находятся значения функции $u_{e,m}^{(n)} = v_{e,m}^{(n)}$ ($n = 0,1,\ldots$ – номер итерации) на ребрах и в макроузлах $\gamma_{e,m}$.

3. Из решения краевой задачи (4) с граничными условиями Дирихле $v = v^{(n)} = v_f^{(n)} \bigcup v_{e,m}^{(n)}$ находятся значения искомой функции на n - ом приближении в подобластях.
4. Рассчитываются производные, входящие в выражение (5).
5. Делается очередной $(n+1)$ -ый шаг по решению уравнения Пуанкаре – Стеклова (2) и находятся значения функции $v_f^{(n+1)}$ на гранях.
6. Если сходимость итерационного процесса достигнута, то находится окончательное решение на ребрах, в макроузлах и в подобластях, если же это не так, то повторяется пункты 2 – 5.

На квазиструктурированной сетке $\bar{\Omega}_h$ краевая задача (4) методом конечных разностей, конечных элементов или конечных объемов заменяется приближенной задачей

$$\Delta_h u_h = g_1, \quad l_h u_h\big|_\Gamma = g_2, \quad u_h\big|_\gamma = v_h, \tag{6}$$

где u_h, v_h – приближенные значения функций u, v, а Δ_h, l_h – аппроксимации оператора Лапласа и оператора граничных условий. Ее решение сводится к решению дискретных подзадач на подсетках $\overline{\Omega}_{h,m}$, что может быть выполнено параллельно. Тем самым мы реализуем на каждой итерации по подобластям этап 3, описанного выше алгоритма.

Для решения подзадач на интерфейсе введем на гранях γ_f сетку ω_f, на ребрах γ_e – сетку ω_e. Для единообразия сетку из макроузлов обозначим как $\omega_m = \gamma_m$. В объединении $\gamma_{e,m}$ будем рассматривать сетку $\omega_{e,m} = \omega_e \bigcup \omega_m$.

Для нахождения функции v_f заменим в (5) производные приближенными разностными соотношениями и потребуем, чтобы разность приближенных производных в узлах сетки ω_f обращалась в нуль, что дает систему линейных алгебраических уравнений (подробнее см. [3])

$$Av_f + b = 0, \tag{7}$$

где A – квадратная матрица, а b, v_f – векторы.

Элементы матрицы A и вектора b не известны. Известно лишь действие оператора F_h, аппроксимирующего оператор F, на какую-либо функцию $\tilde{v}_f$, заданную в узлах ω_f. Это действие определяется формулой

$$F_h \tilde{v}_f = A\tilde{v}_f + b. \tag{8}$$

Для вычисления компонент вектора b дадим $\tilde{v}_f$ пробное значение $\tilde{v}_f^{(0)} = 0$, решим соответствующую краевую задачу, а затем, вычислив $F_h \tilde{v}_f^{(0)}$, получим искомый вектор.

Решение системы линейных алгебраических уравнений (7) будем проводить каким –либо итерационным методом в подпространствах Крылова [7]. Замечательным свойством данных методов, которое мы используем, является то, что они не требуют знания элементов матрицы A, а требуют лишь действия A на некий вектор p, что согласно (8) может быть вычислено как

$$Ap = F_h p - b.$$

На каждом шаге крыловского итерационного процесса необходимо решать краевые подзадачи в подобластях, поэтому его называют итерационным процессом по подобластям. Положительным свойством трудоемкой процедуры итераций по подобластям является то, что она допускает параллельную реализацию.

Решение подзадач на ребрах и в макроузлах составляет этап 2, описанного выше алгоритма и проводится на каждой n-ой итерации по подобластям. К моменту проведения данных вычислений значения функции $v_f^{(n)}$ на гранях считаются известными.

На сетке $\omega_{e,m}$ с привлечением узлов сетки ω_f аппроксимируем исходное уравнение (3). Получим систему сеточных уравнений

$$B\tilde{v} = f \tag{9}$$

с матрицей B относительно функции $\tilde{v}$, определенной на ребрах и в макроузлах. Матрицу B можно представить в блочном виде

$$B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix},$$

и, соответственно, векторы, входящие в (9), как

$$\tilde{v} = \begin{bmatrix} \tilde{v}_e \\ \tilde{v}_m \end{bmatrix}, \quad f = \begin{bmatrix} f_e \\ f_m \end{bmatrix}.$$

Здесь блоки имеют следующие размерности:
$B_{11} : N_e \times N_e$, $B_{12} : N_e \times N_m$, $B_{21} : N_m \times N_e$, $B_{22} : N_m \times N_m$, где N_e – число реберных узлов ω_e, а N_m – число макроузлов $\omega_{m,}$.

Решение системы (9) будем проводить при помощи итерационного процесса, каждый шаг которого состоит из двух полушагов

$$\begin{aligned} &B_{11}\tilde{v}_e^{\nu+1/2} + B_{12}\tilde{v}_m^{\nu} = f_e \\ &B_{21}\tilde{v}_e^{\nu+1/2} + B_{22}\tilde{v}_m^{\nu+1} = f_m, \quad \tilde{v}_e^{\nu+1} = \tilde{v}_e^{\nu+1/2}, \end{aligned} \tag{10}$$

где $\nu = 0,1,\ldots$ – номер итерации.

На первом из них вычисляются значения функции на ребрах при фиксированных значениях в макроузлах, а на втором – корректируются значения в макроузлах при фиксированных значениях на ребрах. На ребрах мы имеем «одномерные» сеточные уравнения, так как известные значения функции $v_f^{(n)}$ на гранях, входящие в аппроксимацию уравнения (3), исключаются и переносятся в правую часть f_e.

Второй полушаг по сути дела состоит в пересчете значений в отдельных макроузлах, которые также не связаны друг с другом, а связаны с полученными на первом полушаге значениями на ребрах.

Рассмотрим решение подзадач в подобластях. Запишем подзадачу вида (6) в m - ой подобласти в матричном виде

$$Du = g, \tag{11}$$

где D – квадратная матица, u – искомый вектор (индексы h, m мы опускаем), g – известный вектор. Рассмотрим аналог итерационного неявного метода Писмана–Речфорда для трехмерного случая. При решении системы (11) он может быть представлен как

$$u^{n-1/2} = u^{n-1} - \omega_z\left(D_{xy}u^{n-1/2} + D_z u^{n-1} - g\right), \quad u^n = u^{n-1/2} - \omega_z\left(D_{xy}u^{n-1/2} + D_z u^n - g\right), \tag{12}$$

где $n = 1,2,\ldots$ – номера итераций, ω_z – итерационный числовой параметр, а D_{xy} и D_z – пятидиагональная и трехдиагональная матрицы такие, что $D = D_{xy} + D_z$ (подробнее см. [7]). Отсюда видно, что на полуцелом шаге решаются «двумерные» системы, а на целом – «одномерные». Решение «двумерных» систем также проводится методом Писмана Рэчфорда вида (12), в котором матрица D_{xy} представляется в виде $D_{xy} = D_x + D_y$, где D_x, D_y – трехдиагональные матрицы. Решение всех «одномерных» систем, к которым приводит реализация алгоритма вида (12), осуществляется методом прогонки.

3. Технологии распараллеливания

При распараллеливании рассматриваемой задачи применяется две парадигмы – MPI-распараллеливание и CUDA-распараллеливание, реализуемые соответственно на вычислительном кластере и вычислительных ускорителях GPU.

3.1 MPI-распараллеливание

MPI-распараллеливанию, в первую очередь, подлежит итерационный процесс по подобластям, который занимает подавляющую часть времени решения всей задачи, и, во вторую очередь, внутренний итерационный процесс (10) поиска решений на ребрах и в макроузлах. Эти итерационные процессы обладают внутренним естественным параллелизмом и не требуют дополнительных вычислительных затрат. Эффективность их распараллеливания целиком и полностью зависит от технологии проведения расчетов, от организации обменов между процессорами вычислительной сети.

Важным звеном в технологической цепи решений по достижению эффективности распараллеливания является отображение сеточных данных на вычислительную сеть. Обычно принятый способ отображения: «одна подобласть – один процессор» в данном случае не эффективен, так как подобласти могут содержать различное число узлов, в которых вычисляются значения искомой функции (в дальнейшем – счетных узлов), что приводит к разбалансировке загрузки процессоров. Поэтому подобласти группируются в объединения, содержащие приблизительно одинаковое число счетных узлов, и устанавливается отображение: «одно объединение – один процессор». Передача информации между подобластями одного объединения обходится без межпроцессорных обменов, а между подобластями различных объединений требуется их проведение. Инициализация и осуществление межпроцессорных обменов являются самыми медленными операциями процесса расчета. В связи с этим в основу технологии MPI-распараллеливания кладутся асинхронные операции, позволяющие хотя бы частично проводить обмены на фоне выполнения необходимых арифметических и логических опреаций.

3.2 CUDA-распараллеливание

Решение подзадач в каждой подобласти осуществляется при помощи трехмерного аналога метода Писмана – Речфорда, который распараллеливается на вычислительных ускорителях GPU.

CUDA – это архитектура параллельных вычислений от NVIDIA [8], позволяющая существенно увеличить вычислительную производительность благодаря использованию графических процессоров GPU (Graphics Processing Unit). GPU состоит из процессорных ядер, которые в терминологии NVIDIA называются Streaming Multiprocessor (SM), каждое из которых выполняет сотни программных нитей. Выполнение нитей на GPU происходит блоками, каждый из которых может состоять от 1 до 1024 нитей. Объединение исполняемых блоков называется гридом (Grid). В CUDA блок можно представить в виде одно-, двух- или трехмерного массива, где индекс массива – это индекс исполняемой нити. Множество блоков в гриде представляется в виде одного двумерного массива.

Программу, реализующую трехмерный аналог метода Писмана – Речфорда, можно условно разделить на три части:

1) Выделение памяти на GPU и копирование данных из памяти CPU.
2) Запуск итерационного процесса на GPU.
3) Копирование результата в память CPU и удаление выделенной памяти на GPU.

Копирование и выделение данных на GPU осуществляется средствами CUDA. Данные, являющиеся константами в итерационном процессе, копируются в константную память устройства до проведения итераций. Блок – схема проведения итерационного процесса показана на рис. 1.

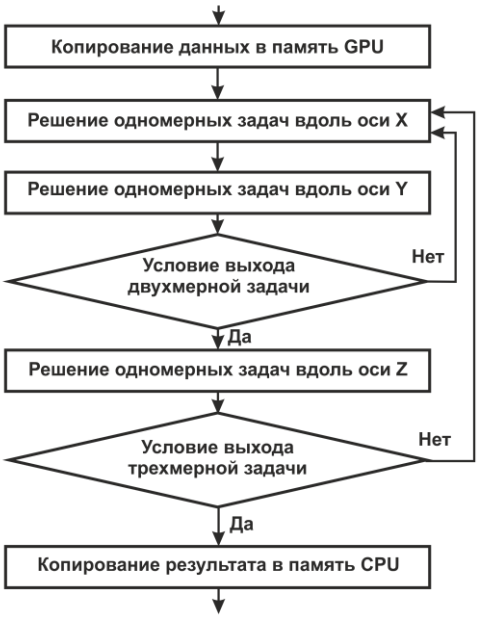

Рис. 1.Блок схема итерационного процесса

Рассматриваемый алгоритм реализован в виде функции языка С. На ее вход подается информация о краевых условиях и правой части исходной краевой задачи, параметры подсетки, на которой решается рассматриваемая в данной подобласти краевая подзадача, счетные параметры:

Работу данной функции можно условно разделить на две части: 1) подготовка и 2) расчет. В первой части вычисляются коэффициенты для метода прогонки; выделяется память на GPU, в которую копируются коэффициенты прогонки, краевые значения и значения правых частей уравнения Пуассона. Во второй части реализован двухуровневый итерационный процесс в виде двухуровневого цикла. Во вложенном цикле на GPU запускаются два расчета одномерных задач вдоль осей OX и OY и расчет невязки двумерной задачи, по окончании которых проверяется условие на выход из цикла. В основном цикле на GPU запускается расчет одномерных задач вдоль оси OZ и расчет невязки трехмерной задачи.

4. Численные эксперименты

Цель проводимых численных экспериментов – исследование эффективности применения графических ускорителей при распараллеливании решения трехмерных краевых задач на квазиструктурированных сетках. Критерием эффективности служила величина отношения времени расчетов с использованием только CPU ко времени расчетов с использованием CPU и GPU.

Рассматривалась следующая модельная задача

$$\Delta u = 0, \quad u\big|_{\Gamma} = 1$$

в единичном кубе $\overline{R}$. Проводилась декомпозиция расчетной области при помощи равномерной макросетки

$$\overline{\Omega}_H = \left\{ X_I = IH_x, Y_J = JH_y, Z_K = KH_z, I = \overline{0, N_x}, J = \overline{0, N_y}, K = \overline{0, N_z}, H_x = \frac{1}{N_x}, H_y = \frac{1}{N_y}, H_z = \frac{1}{N_z} \right\},$$

где N_x, N_y, N_z – заданные целые числа.

В замкнутых макроэлементах $\overline{R}_m = \overline{R}_{I,J,K}$ строились равномерные параллелепипедальные подсетки

$$\overline{\Omega}_{h,m} = \left\{ x_{i_m} = X_I + i_m h_{x,m},\ y_{j_m} = Y_J + j_m h_{y,m}, z_{k_m} = Z_K + k_m h_{z,m} \quad i_m = \overline{0, n_{x,m}},\ j_m = \overline{0, n_{y,m}}, k_m = \overline{0, n_{z,m}} \right\}$$

с шагами $h_{x,m} = \dfrac{X_{I+1} - X_I}{n_{x,m}}, \quad h_{y,m} = \dfrac{Y_{J+1} - Y_J}{n_{y,m}}, \quad h_{z,m} = \dfrac{Z_{K+1} - Z_K}{n_{z,m}}$.

В качестве оператора Δ_h выбирались обычные семиточечные разностные операторы Лапласа. Итерации по подобластям проводились при помощи метода сопряженных градиентов, легко поддающегося распараллеливанию.

Вычисления проводились на кластере НКС-30Т Сибирского суперкомпьютерного центра (ССКЦ СО РАН, г. Новосибирск), который имеет следующие характеристики одного вычислительного узла:

- CPU:
  2 CPU Xeon X5670 6 x 2.93 GHz
- GPU:
  3 NVIDIA Tesla M 2090 6Gb на архитектуре Fermi
  512 CUDA Cores

Проведенные серии экспериментов тематически можно представить в виде трех разделов, приведенных ниже.

4.1 Исследование эффективности в зависимости от числа ядер CPU

В таблице 1 приведено отношение времени, затраченного на вычисление решения только на CPU ядрах к времени, затраченному на вычисление решения с использованием GPU. Макросетка имела параметры $N_x, N_y = 4, N_z = 3$, а параметры согласованных подсеток приведены в таблице 1. При запуске расчета на GPU устройстве размер блока брался равным 8x8, что позволило обеспечить эффективное использование параметров GPU.

Из полученных результатов можно сделать следующие выводы: 1) ускорение, полученное с использованием GPU, с точностью до погрешностей измерения не зависит от количества ядер CPU, 2) рост ускорения с ростом числа узлов в сетке связан с возрастающим объемом вычислений, приходящихся на одну нить GPU устройства.

Таблица 1. Ускорение в зависимости от количества ядер CPU

Ядра CPU	Сетка				
	16^3	32^3	48^3	64^3	128^3
1	5	20	39	33	45
3	4	19	37	34	47
6	4	18	37	34	47
12	4	19	38	36	47

4.2 Исследование эффективности на согласованных и несогласованных квазиструктурированных сетках

В проводимых экспериментах макросетка имела параметры $N_x, N_y = 4, N_z = 3$, а размеры подсеток по одному направлению приведены на рис.2. Несогласованная сетка была такой, что любые две соседние подсетки имели различное, отличающееся в два раза по одному направлению, число узлов. На рис. 2 дано число узлов густых подсеток в несогласованных сетках.

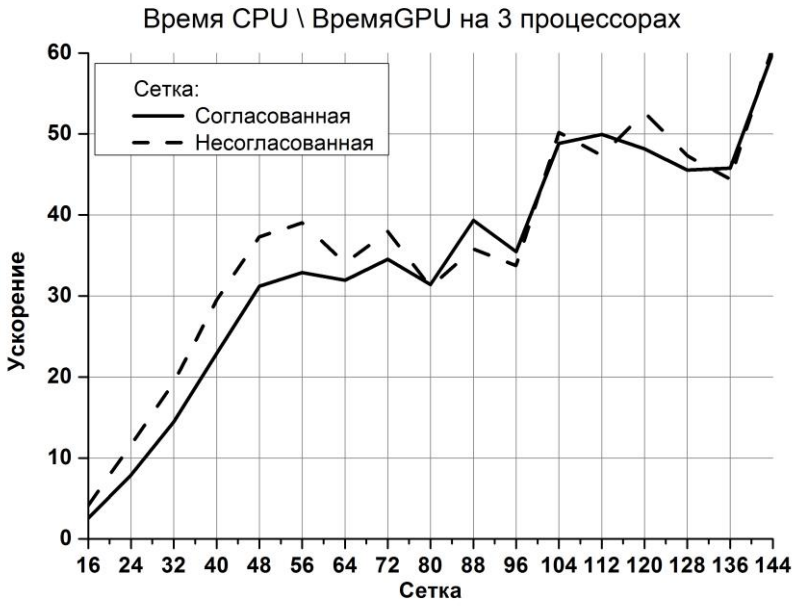

Рис. 2. Ускорение, полученное на различных сетках на 3 ядрах CPU

Из приведенного рисунка следует, что эффективность применения GPU незначительно отличается для согласованных и несогласованных сеток. Количество блоков, вычисляемое на GPU, зависит от размера сетки. Если это количество не кратно количеству блоков, которое GPU устройство может вычислять одновременно, то это ведет к снижению эффективности вычисления. Данное свойство объясняет нелинейность ускорения, изображенного на рис. 2.

4.3 Исследование эффективности в зависимости от размеров макросетки

На рис. 3 показана зависимость полученного ускорения в зависимости от размера сетки в подобласти, при различных размерах макросетки. Различие в ускорениях объясняется тем, что, с ростом количества узлов макросетки, растет время на обмен и обработку данных между подобластями на одном ядре CPU.

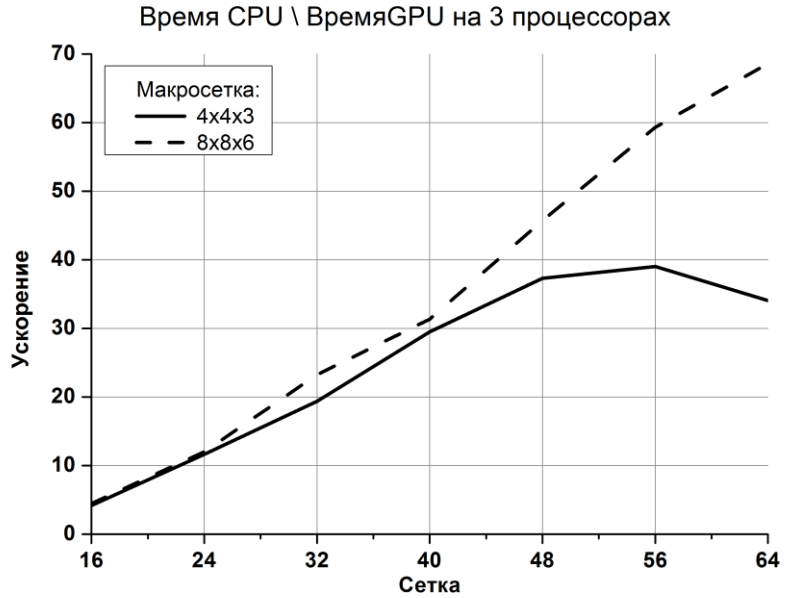

Рис. 3. Ускорение при разном количестве узлов макросетки

5. Заключение

В работе исследована эффективность применения графических ускорителей при распараллеливании решения трехмерных краевых задач на квазиструктурированных сетках. Распараллеливание осуществляется методом декомпозиции расчетной области на подобласти, сопрягаемые без наложения, основанном на прямой конечно-разностной аппроксимации уравнения Пуанкаре – Стеклова на интерфейсе. Возникающий при этом итерационный процесс по подобластям распараллеливается на CPU. Для решения подзадач в подобластях, составляющего наибольшую часть времени решения всей задачи, применяется метод Писмана – Рэчфорда, обладающий быстрой сходимостью. Его распараллеливание осуществляется на GPU в системе CUDA. Показано, что применение графических ускорителей значительно (более 60 раз) сокращает время решения задачи по сравнению с расчетами только на CPU.

Литература

1. Василевский Ю.В., Ольшанский М.А. Краткий курс по многосеточным методам и методам декомпозиции области. М.: МГУ. 2007.
2. Quarteroni A., Valli A. Domain Decomposition Methods for Partial Differential Equations // Oxford: Clarendon Press. 1999.
3. Свешников В.М. Построение прямых и итерационных методов декомпозиции // СибЖИМ. 2009. Т.12, № 3(39). С. 99 – 109.
4. Свешников В.М., Беляев Д.О. Построение квазиструктурированных локально-модифицированных сеток для решения задач сильноточной электроники // Вестник ЮУрГУ. 2012. № 40(299). С. 130 – 140.
5. Свешников В.М., Рыбдылов Б.Д. О распараллеливании решения краевых задач на квазиструктурированных сетках // Вестник ЮУрГУ, серия ВМИ. 2013. Т.2, № 3. С. 63 – 72.
6. Корнеев В.Д. Параллельное программирование в MPI // Новосибирск. ИВМиМГ СО РАН. 2002.
7. Ильин В.П. Методы конечных разностей и конечных объемов для эллиптических уравнений. // Новосибирск: ИВМиМГ (ВЦ) СО РАН, 2001.
8. NVIDIA. CUDA C Best Practices Guide. 2012.

Studying the effectiveness of graphics accelerators paralleling solutions of three-dimensional boundary value problems on quasi-structured grids

I.A. Klimonov[2], V.D. Korneev[1], V.M. Sveshnikov[1,2]

Institute of Computational Mathematics and Mathematical Geophysics SB RAS[1],
Novosibirsk State University[2]

When paralleling solutions of three-dimensional boundary value problems on quasi-structured grids by decomposition of the computational domain into subdomains, matched without the imposition of the most time consuming computational procedure is a solution of subproblems in subdomains. Using the parallelepiped quasi-structured grids makes possible to use for this purpose rapidly convergent method of alternating directions. The parallelization of the iterative process on subdomains is carried out in CPU and to solve the subproblems in this paper we propose to use the GPU graphics accelerators. Experimental studies of the use of graphics accelerators for solving subproblems by Peaceman – Rechford method is presented. We give experimental estimates of acceleration parallelization in hybrid computing environment CPU + GPU as compared to the calculations only on the CPU.

*Keywords***:** boundary value problem, domain decomposition method, Poincare – Steklov equation, quasi-structured grids, Peaceman – Rechford method, graphic accelerators.

References

1. Vasilevskiy Yu.V., Ol'shanskiy M.A. Kratkiy kurs po mnogosetochnym metodam i metodam dekompozitsii oblasti [Short course on multigrid methods and domain decomposition methods]. Moscow. Publishing of the Moscow State University. 2007.
2. Quarteroni A., Valli A. Domain Decomposition Methods for Partial Differential Equations // Oxford: Clarendon Press. 1999.
3. Sveshnikov V.M. Postroenie pryamykh i iteratsionnykh metodov dekompozitsii [Construction of Direct and Iterative Decomposition Methods] // Sibirskiy zhurnal industrial'noy matematiki [Siberian Journal of Industrial Mathematics]. 2009. No. 3(39). Vol.12. P. 99 – 109.
4. Sveshnikov V.M., Belyaev D.O. Postroenie kvazistrukturirovannykh lokal'no-modifitsirovannykh setok dlya resheniya zadach sil'notochnoy elektroniki [Construction of quasi-structured locally modified grids to solve the problems of High Current Electronics]// Vestnik Yuzho-Uralskogo gosudarstvennogo universiteta. Seriya "Matematicheskoe modelirovanie i programmirovanie" [Bulletin of South Ural State University. Series: Mathematical Modeling, Programming & Computer Software]. 2012. No. 40(299). P. 130 – 140.
5. Sveshnikov V.M., Rybdylov B.D. O rasparallelivanii resheniya kraevykh zadach na kvazistrukturirovannykh setkakh [On parallel solution of boundary value problems on quasistructured grids]// Vestnik Yuzho-Uralskogo gosudarstvennogo universiteta. Seriya " Vychislitel'naya matematika i informatika" [Bulletin of South Ural State University. Series: Computational mathematics and Informatics]. 2013. No. 3. Vol..2,. P. 63 – 72.
6. Korneev V.D. Parallel'noe programmirovanie v MPI [Parallel programming in MPI] Novosibirsk. Publishing of the Institute of Computational Mathematics and Mathematical Geophysics Siberian Branch of Russian Academy of Sciences . 2002.
7. Il'in V.P. Metody konechnykh raznostey i konechnykh ob"emov dlya ellipticheskikh uravneniy. [The method of finite differences and finite volumes for elliptic equations] Novosibirsk. Publishing of the Institute of Computational Mathematics and Mathematical Geophysics Siberian Branch of Russian Academy of Sciences. 2001.
8. NVIDIA. CUDA C Best Practices Guide. 2012.