Параллельные алгоритмы решения краевых задач в подобластях в методе декомпозиции с использованием гибридных вычислений GPU+CPU*

А.В. Медведев
Институт Вычислительных Технологий СО РАН
e-mail: andertsk@gmail.com

И.Ю. Турчановский
e-mail: tur@hcei.tsc.ru

В.М. Свешников
Институт вычислительной математики и математической геофизики СО РАН
e-mail: victor@lapasrv.sscc.ru

Проведено распараллеливание метода последовательной верхней релаксации для решения сеточных уравнений, возникающих при аппроксимации краевых задач на квазиструктурированных сетках, с использованием графических ускорителей GPU. Показано, что быстродействие программы с использованием GPU в рамках текущей реализации по отношению к аналогичной программе на CPU, увеличивается в 25.2 раз, используя последовательный код на CPU и в 8.3 раз используя параллельный код на CPU (4 ядра).

1. Введение

При решении краевых задач на квазиструктурированных сетках [1] в итерационном методе декомпозиции по подобластям на каждом шаге необходимо решать краевые подзадачи. Декомпозиция на подобласти происходит путем построения макросетки, а в каждой подобласти строится своя микросетка. Макросетка и микросетки образуют результирующую квазиструктурированную сетку. Цель настоящей работы состоит в отображении алгоритмов декомпозиции, рассмотренных в работе [2], на гибридную архитектуру CPU+GPU. Суть предлагаемого подхода заключается в автономной аппроксимации исходного дифференциального уравнения в подобластях и уравнения Пуанкаре–Стеклова на границе сопряжения подобластей (интерфейсе). Для решения системы линейных алгебраических уравнений, аппроксимирующих уравнение Пуанкаре–Стеклова используется итерационный метод сопряженных градиентов [3], а для решения подзадач в подобластях был выбран метод последовательной верхней релаксации [4], который обладает внутренним параллелизмом, легко отображаемым на GPU (более подробно см. работу [5]). Сравнение быстродействия проводилось между тремя версиями кода:

1. последовательный код на CPU,

*Работа выполнена при финансовой поддержке Российского фонда фундаментальных исследований (проект № 12-01-00076) и СО РАН (интеграционные проекты №№ 104, 126, 130).

2. параллельный код на CPU,

3. гибридный код: параллельный код на CPU + код GPU.

Были проведены численные эксперименты на модельной задаче, которые показали, что быстродействие варианта 3 по отношению к варианту 1 увеличивается в 25.2 раза, а по отношению к варианту 2 – в 8.3 раза.

2. Постановка краевой задачи

В замкнутой двумерной области $\overline{G} = G \cup \Gamma$ с границей Γ требуется решить краевую задачу:

$$\Delta u = g_1,\ lu|_\Gamma = g_2,$$

где $u = u(T)$ – искомая функция; $g_1 = g_1(T),\ g_2 = g_2(T)$ – заданные функции ($T = (x, y)$ – текущая точка; x, y – декартовы координаты); Δ – оператор Лапласа; l – оператор граничных условий. Решение данной задачи ищется на квазиструктурированной сетке методом декомпозиции.

3. Построение квазиструктурированных сеток

Вокруг расчетной области $\overline{G}$ описывается прямоугольник $R = \{0 \le x \le D_x,\ 0 \le y \le \le D_y\}$, где D_x, D_y – заданы. В R строится равномерна *макросетка*: $\Omega_H = \{X_I = = IH_x;\ Y_J = JH_y;\ I = \overline{0, N_x};\ J = \overline{0, N_y};\ H_x = \frac{D_x}{N_x};\ H_y = \frac{D_y}{N_y}\}$, где N_x, N_u – заданные числа, H_x, H_y – шаги, которые намного превосходят шаги сетки, на которой аппроксимируется исходная задача.

Таким образом, исходная область фактически подвергается декомпозиции на непересекающиеся подобласти. Все подобласти делятся на три типа: внутренние, содержащие только точки G, граничные, содержащие точки G и Γ, а также внешние, не содержащие точек G. В дальнейшем внешние подобласти исключаются из расчетов.

Точки пересечения координатных линий образуют *макроузлы*. Координатные линии макросетки являются границами сопряжения подобластей или *интерфейсом*.

После того как построена макросетка, внутри каждой подобласти строится своя сетка (*микросетка*) $\overline{\Omega}_{h,k} = \{x_{i_k} = X_I + i_k h_{x,k},\ y_{j_k} = Y_J + j_k h_{y,k},\ i_k = \overline{0, n_{x,k}},\ j_k = = \overline{0, n_{y,k}}\}$, с шагами $h_{x,k} = \frac{X_{I+1} - X_I}{n_{x,k}};\ h_{y,k} = \frac{Y_{J+1} - Y_J}{n_{y,k}}$, где $n_{x,k}, n_{y,k}$ – заданные целые числа. Подчеркнем, что величины $n_{x,k}, n_{y,k}$, которые определяют плотность узлов в подсетках, задаются в зависимости от особенностей решаемой задачи, то есть подсетки могут быть несогласованными.

На интерфейсе строится своя сетка ω_h, которая состоит из узлов подсеток $\overline{\Omega}_{h,k}$. На рис.1 изображен фрагмент квазиструктурированной сетки, состоящий из четырех подобластей (черные кружки обозначают макроузлы).

4. Технология проведения расчетов

Для нахождения решения исходной краевой задачи на квазиструктурированной сетке применяется метод декомпозиции расчетной области на непересекающиеся подобласти.

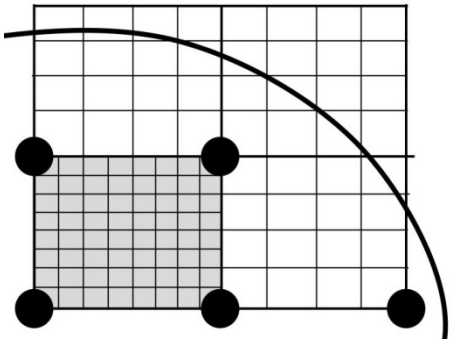

Рис. 1: Фрагмент квазиструктурированной сетки

Сам метод декомпозиции и технология его применения к квазиструктурированным сеткам изложены в работах [2, 5]. Суть рассматриваемого подхода заключается в проведении итераций по подобластям или макроэлементам квазиструктурированной сетки. На каждой такой итерации необходимо решать краевые задачи Дирихле во внутренних и граничных подобластях и вычислять суммы внутренних нормальных производных на интерфейсе. Распараллеливание самого итерационного процесса по подобластям проводится на CPU в рамках системы MPI [6]. Распараллеливание решения краевых задач осуществляется на GPU в рамках системы CUDA [7]. В целом технология проведения расчетов выглядит следующим образом:

1. На CPU на каждом шаге итерационного процесса по подобластям вычисляются значения искомой функции в узлах сетки ω_h на интерфейсе.

2. Эти значения копируются в память GPU.

3. На GPU методом ПВР решаются краевые задачи в подобластях и на сторонах вычисляются внутренние нормальные производные.

4. Полученные значения на сторонах подобластей копируются в память CPU.

5. Пересчитываются значения на интерфейсе и описанный процесс повторяется, начиная с шага 2, пока он не сойдется с заданной точностью.

5. Численные эксперименты

Численные эксперименты проводились на следующей тестовой задаче

$$\Delta u = 0,\ u|_\Gamma = g,$$

в квадрате $\overline{G} = \{1 \le x \le 1.5,\ 0 \le y \le 0.5\}$. Данная задача представляла собой фрагмент более общей задачи о решении уравнения Лапласа в области, ограниченной двумя концентрическими окружностями с радиусами $R_1 = 1$, $R_2 = 2$, на которых задаются значения искомой функции $\overline{u}$, равные соответственно 0, 1. Последняя задача имеет точное решение:

$$\overline{u} = ln\frac{r}{R_1}/ln\frac{R_2}{R_1},$$

где r – радиус-вектор. Функция g выбиралась, как $g = \overline{u}|_\Gamma$, и тогда точное решение исходной задачи есть $u = \overline{u}$ в G.

Итерации проводились методом сопряженных градиентов [3]. Критерием останова было уменьшение относительной невязки до заданной величины ε. Целью проводимых экспериментов было установить ускорение вычислений с использованием гибридной схемы вычислений по отношению к последовательным и параллельным расчетам на CPU. Для этого была написана программа на языке программирования CUDA C. Расчетные эксперименты проводились на оборудовании Intel Core i5, NVIDIA GPU Tesla C2070 в среде linux CentOS 6.2.

Ниже приведено два графика зависимости ускорения от N – числа подобластей в одном направлении (на рис. 2 – ускорение программы с использованием GPU по отношению к однопоточной программе, а на рис. 3 по отношению к многопоточной – 4 потока). В силу аппаратных особенностей GPU, для типа данных *double* максимально допстимой точностью является $\varepsilon = 1.0e - 14$. Эти значения, в дальнейшем, и были взяты в качестве условия остановки итерационного процесса верхней релаксации.

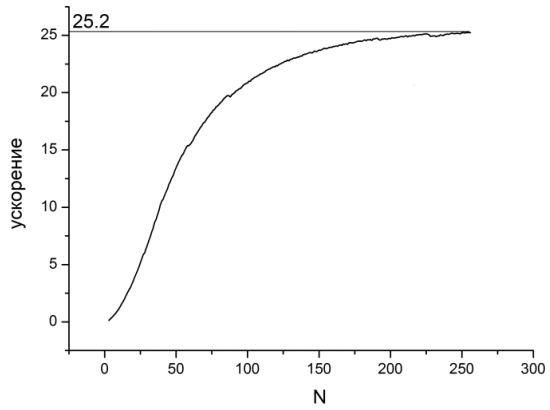

Рис. 2: График ускорения вычислений по отношению к однопоточному коду

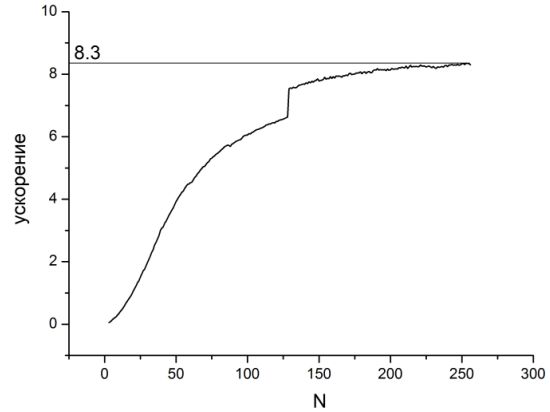

Рис. 3: График ускорения вычислений по отношению к многопоточному коду

Квазиструктурированная сетка состояла из подсеток, каждая из которых содержат $32 * 32$ узла, что обусловлено тем, что на каждом мультипроцессоре именно 32 дискретных вычислителя, а потоки (нити) исполняются группами по 32. Макросетка имела параметры $N_x = N_y = N$, причем N изменялось в пределах $3, \ldots, 256$.

Поведение графиков обусловлено тем, что, начиная с некоторого N, на GPU становятся задействованными практически все процессорные ресурсы, т.е. каждый мультипроцессор загружен на $\approx 100\%$, после чего значение ускорения стабилизируется. Объяснение резкому скачку между $N = 128$ и $N = 129$ на рис. 3 пока не найдено.

Время расчетов с использованием GPU для $N = 256$ и $n = 32$ (количество узлов квазиструктурированной сетки равно $256 * 256 * 32 * 32 \approx 67$млн.) составило 51.38 сек.

6. Заключение

В настоящей работе получены следующие основные результаты

1. Проведена адаптация итерационного метода декомпозиции по подобластям для распараллеливания расчетов во внутренних подобластях на GPU.

2. Разработана программа на языке CUDA C, решающая вышеупомянутую краевую задачу в квадрате размерами $N * N$ подобластей, каждая из которых содержит $n * n$ узлов.

3. Получено, что быстродействие программы с использованием GPU в рамках текущей реализации по отношению к аналогичной программе на CPU, увеличивается в 25.2 раз, используя последовательный код на CPU и в 8.3 раз используя параллельный код на CPU (4 ядра).

Список литературы

[1] Беляев Д.О., Свешников В.М. Построение квазиструктурированных локально-модифицированных сеток для решения задач сильноточной электроники // Вестник ЮУрГУ. 2012. № 40(299). С. 118–128.

[2] Свешников В.М. Построение прямых и итерационных методов декомпозиции // Сиб. Журн. Идустр. Матем.. 2009. Т. 12, № 3(39). С. 99–109.

[3] Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods. / R. Barrett, M. Berry, T. F. Chan, and J. Demmel, et al., Philadelphia: 1994. SIAM. pp. 12–15.

[4] Ильин В.П. Методы конечных разностей и конечных объемов для эллиптических уравнений. Новосибирск: Издательство Института математики, 2000. 345 с.

[5] Медведев А.В., Свешников В.М., Турчановский И.Ю. Распараллеливание решения сеточных уравнений на квазиструктурированных сетках с использованием графических ускорителей. [Электронный ресурс]// XIV Российская конференция с участием иностранных ученых «Распределенные информационные и вычислительные ресурсы» - DICR-2012 (Новосибирск, Россия, 26.11 – 30.11.2012): Материалы конференции. – Новосибирск: ИВТ СО РАН, 2012. – Рег.номер 0321300118. – Режим доступа: http://conf.nsc.ru/dicr2012/ru/reportview/140603 – (1.02.2013)

[6] Корнеев В.Д. Параллельное программирование в MPI. 2002. Новосибирск. ИВМиМГ СО РАН

[7] NVIDIA. CUDA C Best Practices Guide. 2012.