КОМПЬЮТЕРНЫЕ НАУКИ

УДК 004.94: 004.272.2: 519.612: 519.63

© *С. П. Копысов, А. К. Новиков, Ю. А. Сагдеева*

РЕШЕНИЕ СИСТЕМ УРАВНЕНИЙ МЕТОДА ГАЛЁРКИНА С РАЗРЫВНЫМИ БАЗИСНЫМИ ФУНКЦИЯМИ НА ГРАФИЧЕСКОМ УСКОРИТЕЛЕ[1]

Рассматриваются особенности решения систем уравнений метода Галёркина с разрывными функциями на графических процессорах GPU прямым методом и методами подпространств Крылова с различными предобуславливателями. Производительность программной реализации решения систем на GPU сравнивается с аналогичной, полученной на многоядерном процессоре CPU.

Ключевые слова: метод Галёркина с разрывными базисными функциями, системы линейных алгебраических уравнений, методы подпространств Крылова, предобуславливатель, разрежённые матрицы, вычисления общего назначения на графических устройствах.

Введение

Метод Галёркина с разрывными базисными функциями (далее разрывный метод Галёркина или РМГ) [1] пользуется всё большей популярностью при решении уравнений в частных производных, в частности, из-за его гибкости в использовании расчётных сеток (несогласованные сетки, h- и hp-адаптация) и потенциально высокой эффективности распараллеливания этапа формирования систем уравнений [2–5]. В качестве основного недостатка РМГ необходимо выделить больший размер получаемых систем уравнений по сравнению с классическим методом Галёркина (МГ), которые кроме того являются достаточно плохо обусловленными. Решение таких систем требует значительных вычислительных ресуров. В работе рассматриваются подходы к распараллеливанию этапа решения систем уравнений РМГ и вычислениям на графических ускорителях.

Современные вычислительные системы, как правило, содержат кроме многоядерного центрального процессора мультиядерные графические процессоры (GPU — Graphics Processing Units), которые могут обеспечить существенное увеличение производительности при вычислениях общего назначения. Гетерогенность такой вычислительной системы учитывается за счёт разбиения кода программ на часть, исполняемую на центральном процессоре и часть, выполняемую на графических процессорах. В этом случае, реализуется модель параллелизма данных (Data Parallel) [6], когда множеством процессорных устройств GPU обрабатываются различные блоки данных. Появление в графических ускорителях аппаратной поддержки вычислений с двойной точностью позволило решать плохо обусловленные системы уравнений РМГ.

В данной работе сравнивается решение систем разрывного метода Галёркина на графическом ускорителе прямым методом и методами подпространств Крылова (методом сопряженных градиентов CG, методом обобщённых минимальных невязок GMRES и методом бисопряженных градиентов BiCGStab) с различными видами предобуславливания [7]. Вычислительные затраты при решении систем РМГ на GPU сопоставляются с аналогичными, полученными на многоядерном процессоре (CPU), и при решении систем классического метода Галёркина (как при близких размерах систем, так и при одинаковых расчётных сетках).

[1]Работа выполнена при финансовой поддержке РФФИ (грант 09–01–00061) и РЦП УрО РАН.

§ 1. Разрывный метод Галёркина для задачи теории упругости

Рассмотрим задачу теории упругости в ограниченной области $\Omega \in \mathbb{R}^2$

$$\nabla\sigma + g = 0 \text{ в } \Omega; \ \sigma = C : \nabla^s u \text{ в } \Omega; \ u = u^D \text{ на } \Gamma^D; \ \sigma n = f^N \text{ на } \Gamma^N, \tag{1.1}$$

где C — тензор, связывающий напряжения σ и деформации ε; n — вектор нормали к Γ^N, $u : \Omega \to \mathbb{R}^2$ — поле перемещений, $g : \Omega \to \mathbb{R}^2$ — объемные силы.

Будем решать задачу (1.1) с помощью РМГ. Пусть $\tilde{\Omega}$ — разделение области Ω на M элементов K_i, таких что $\cup_{i=1}^{M} K_i = \tilde{\Omega}$, $K_i \cap K_j = \emptyset$. Совокупность внутренних ребер между элементами обозначим как $\tilde{\Gamma}$. Внешнюю границу области Ω обозначим как Γ. Граничные условия Дирихле заданы на Γ^D, а условия Неймана на Γ^N, так что $\Gamma^D \cup \Gamma^N = \Gamma$, $\Gamma^D \cap \Gamma^N = \emptyset$. Определим оператор среднего $\langle\cdot\rangle$ на ребре, принадлежащем двум конечным элементам $\langle u^h\rangle := \frac{1}{2}\left(u_1^h + u_2^h\right)$ на $\tilde{\Gamma}$, $\langle u^h\rangle := u^h$ на Γ, где нижние индексы 1 и 2 относятся к конечным элементам, имеющим общее внутреннее ребро. Оператор скачка $[\![\cdot]\!]$ определим как $[\![u^h]\!] := u_1^h - u_2^h$ на $\tilde{\Gamma}$, $[\![u^h]\!] := u^h$ на Γ. Запишем вариационную формулировку для задачи теории упругости по методу внутреннего штрафа [1, 8]: найти функцию $u^h \in U^h$ такую, что для всех $\omega^h \in W^h$

$$\int_{\tilde{\Omega}} \nabla^s\omega^h : C : \nabla^s u^h \, d\Omega + \theta_{DG} \int_{\tilde{\Gamma}} \langle \nabla^s\omega^h : C\rangle n_K \cdot [\![u^h]\!] \, d\Gamma - \\ - \int_{\tilde{\Gamma}} [\![\omega^h]\!] \cdot \langle C : \nabla^s u^h\rangle n_K \, d\Gamma + \int_{\tilde{\Gamma}} [\![\omega^h]\!] \eta [\![u^h]\!] \, d\Gamma = \int_{\tilde{\Omega}} \omega^h \cdot g \, d\Omega + \int_{\Gamma^N} \omega^h \cdot f^N \, d\Gamma, \tag{1.2}$$

здесь u^h — пробная функция; $\eta \geqslant C_1\delta/h^\beta$ — штраф в скачке перемещений $[\![u^h]\!]$ между элементами ($C_1 = \text{const}$); $\beta \geqslant 1/(d-1)$ — параметр, зависящий от размерности задачи; θ_{DG} — параметр, определяющий метод штрафа: $\theta_{DG} = -1$ симметричный РМГ с внутренним штрафом (СРМГ); $\theta_{DG} = +1$ несимметричный РМГ с внутренним штрафом (НРМГ). Левая часть соотношения (1.2) содержит четыре слагаемых: первое слагаемое имеет такой же вид, как в классическом методе Галёркина; второе — определяет симметричность вариационной формулировки ($\theta_{DG} = -1$); третье — обеспечивает согласованность и четвертое — устойчивость ($\delta = 1$).

После перехода от вариационной формулировки (1.2) к конечно-элементной аппроксимации для четырехугольных билинейных элементов [8] и исключения тривиальных уравнений при задании граничных условий Дирихле, получим систему уравнений

$$Au = f, \tag{1.3}$$

где $A \in \mathbb{R}^{N\times N}$ — редуцированная матрица жесткости. Полученная матрица жесткости РМГ имеет разреженную структуру и положительно определена. Для СРМГ матрица симметричная и несимметричная для НРМГ.

Существующие теоретические оценки обусловленности для метода штрафа разрывного метода Галёркина показывают, что число обусловленности имеет порядок от $\mathcal{O}(h^{-2})$ до $\mathcal{O}(h^{-(2p+2)})$ в зависимости от варианта РМГ [9] и порядка аппроксимирующих функций p. Параметр штрафа в свою очередь зависит от шага сетки и может достигать больших значений для получения устойчивого решения. Таким образом, система (1.3) получается плохо обусловленной и для её решения как итерационными, так и прямыми методами необходимо применять вычисления с двойной точностью. Кроме того, для ускорения сходимости итерационных методов система (1.3) преобразуется к виду

$$MAu = Mf, \tag{1.4}$$

здесь M — предобуславливатель.

Для МГ размер системы уравнений (1.4) зависит от числа узлов расчётной сетки, а размер системы уравнений для РМГ пропорционален числу конечных элементов в сетке. Для получения тестовых матриц A рассматривалась плоская задача теории упругости для круговой области радиуса $b = 200$, содержащей включение радиуса $a = 5$. Материалы включения и матрицы упругие, со следующими константами: $E_1 = E_2 = 1$, $\nu_1 = 0.3$ и $\nu_2 = 0.25$. Условия Дирихле задавались на внутренней границе между двумя материалами, как $u_r = a$ и $u_\theta = 0$. Расчётные сетки — квазиструктурированные с неравномерным шагом, уменьшающимся к границе включения и четырехугольной формой элементов. Нумерация узлов и элементов в сетке задавалась произвольным образом. Считалось, что для системы уравнений вида (1.4) сформирована глобальная матрица и вектор правой части.

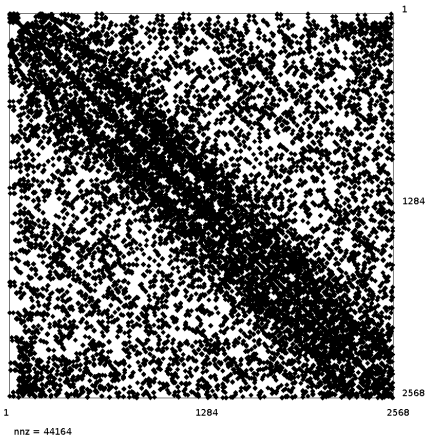

Рис. 1. Портрет матрицы в МГ ($N_{nz} = 44164$)

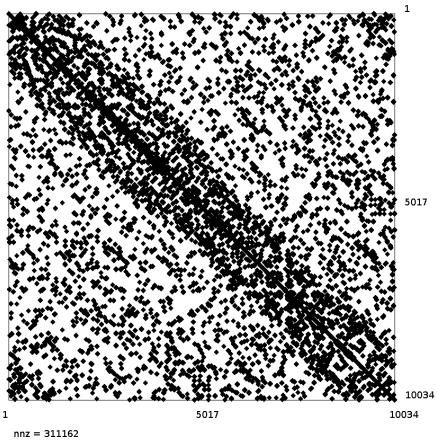

Рис. 2. Портрет матрицы в СРМГ ($N_{nz} = 311162$)

Например, для одной и той же расчётной сетки, содержащей $N_e = 1296$ конечных элементов, портреты глобальных матриц для методов МГ и СРМГ приведёны на рис. 1, 2. Число уравнений в первом случае (МГ) составило $N = 2568$, число ненулевых элементов в матрице $N_{nz} = 44164$, а для СРМГ соответственно — $N = 10034$ и $N_{nz} = 311162$.

Отметим, что кроме большего размера системы СРМГ, матрица имеет более существенную разряженность. Число ненулевых элементов матрицы СРМГ имеет верхнюю оценку $N_{nz} \leqslant N_e s(p+1)^{2d}$; N_e — число конечных элементов сетки; d — размерность пространства элементов; s — шаблон конечного элемента.

Для сравнения шаблоны вкладов двух четырехугольных элементов с билинейной аппроксимацией перемещений в матрицу системы уравнений для первого слагаемого из левой части (1.2) показаны на рис. 3, а остальные слагаемые — на рис. 4.

В отличии от случая, когда структура матрицы известна сразу или может быть определена из физических свойств задачи, шаблон РМГ и вклады в матрицу жесткости отдельных слагаемых существенно осложняют выбор стандартных и построение новых предобуславливателей.

§ 2. Распараллеливание метода решения

Итерационные методы на подпространствах Крылова с предобуславливанием являются наиболее эффективной группой итерационных методов, применяемой для решения разреженных систем линейных алгебраических уравнений (СЛАУ). В этих методах наиболее затратной операцией является произведение матрицы на вектор невязки или направления, которое обладает большой степенью параллелизма. В свою очередь, разрежённая структура матриц РМГ требует при распараллеливании данной операции, приведения её в соответствие с компактными схемами хранения матриц (форматы COO, CRS, ELL, BCRS2 и другие) и оптимизации

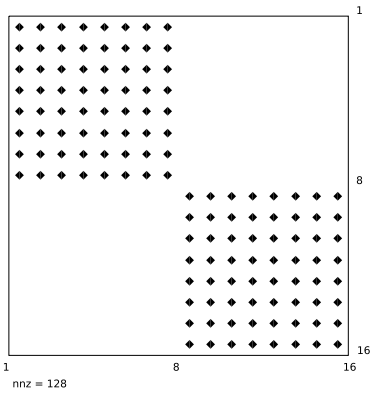

Рис. 3. Портрет вкладов от первого слагаемого в матрицу РМГ

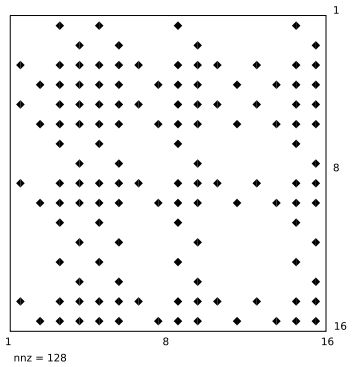

Рис. 4. Портрет вкладов от слагаемых по рёбрам конечных элементов

под архитектуру памяти графического ускорителя. Операции над векторами (скалярные произведения, линейные комбинации, нормы) эффективно выполняются на GPU стандартным программным обеспечением, таким, как библиотека линейной алгебры CUBLAS [10].

Плохая обусловленность систем РМГ предполагает применение предобуславливания. В этом случае распараллеливанию подлежит как процесс построения предобуславливателя, так и его применение. Предобуславливатель может быть построен параллельно на нескольких ядрах CPU или нескольких CPU (M_{MCPU}) или параллельно на GPU (M_{GPU}), а также в ряде случаев последовательно на CPU (M_{CPU}). По способу применения в данной работе рассматривается явное $z_i \leftarrow M^{-1} r_i$ и неявное предобуславливание, при котором решается система $Mz_i = r_i$, здесь r_i — вектор невязки на шаге i. Для GPU эффективно распараллеливается явное предобуславливание. В свою очередь, неявное предобуславливание может выполняться последовательно на CPU, параллельно на нескольких (ядрах) CPU или параллельно на GPU.

Другая группа методов, используемая в работе, относится к классу прямых методов, позволяющих получить приближенное решение системы (1.3) за известное число арифметических операций. Применение в этих методах арифметики с двойной точностью обеспечивает допустимую погрешность решения системы РМГ. Параллельные алгоритмы прямых методов обычно строятся на применении переупорядочивания неизвестных и блочном или фронтальном подходе. Тем не менее, прямые методы обладают сравнительно небольшим потенциалом распараллеливания, а эффективное использование GPU предполагает запуск большого числа параллельных вычислительных процессов (потоков). Поэтому в данной работе разложение небольших матриц вычисляется последовательно, а решение систем с треугольными матрицами осуществляется параллельными алгоритмами.

Конечно же, интерес представляют и гибридные методы, сочетающие достоинства прямых и итерационных методов. К гибридным методам могут быть отнесены: прямые методы с итерационным уточнением; итерационные методы с предобуславливанием на основе неполного разложения A (ILU/CG, IC/CG); методы декомпозиции [11, 12], например, метод дополнения Шура (прямой метод применяется в подобластях, итерационный метод — на границе подобластей). Поэтому рассматриваемые здесь методы и их реализации будут в дальнейшем основой для построения перспективных гибридных решателей СЛАУ.

§ 3. Программная реализация методов решения для графических ускорителей

Методы решения систем уравнений программно реализованы в рамках объектно-ориентированной технологии на C++ и CUDA.

В программном коде выделяется головная часть (головная подпрограмма метода решения), написанная на C++ и выполняемая на CPU и части, выполняемые на графическом устройстве

(операции на матрицей и векторами), написанные на расширении языка C для CUDA.

Головная часть управляет последовательностью шагов алгоритма, например, для предобусловленного метода сопряженных градиентов — это предобуславливание, вычисление последовательности векторов приближенного решения, невязки и A-сопряженного направления, проверка условия выхода из алгоритма. Сами шаги, за исключением операций со скалярными величинами, реализованы в виде kernel-функций CUDA и обращений к функциям библиотеки CUBLAS. При исполнении программы kernel-функции вызываются с CPU, но исполняются на GPU, одновременно каждым потоком (thread-ом). Потоки, выполняемые на GPU характеризуются существенно меньшими затратами на создание, управление и удаление, чем потоки на CPU. Одновременным выполнением множества потоков на GPU обеспечивается массовый параллелизм вычислений. Головная часть управляет вычислениями на GPU, следующим образом:

- Выделяется память на GPU для хранения матрицы и векторов.
- Копируются данные из памяти CPU в выделенную память GPU.
- Запускается kernel (`имя функции <<<параметры запуска>>>(список аргументов)`) — например, подпрограмма вычисления произведений матрицы на вектор.
- Копируется вектор решения из памяти GPU в память CPU.
- Освобождается выделенная память GPU.

В исходном коде части, выполняемой на GPU, определяются вычисления, осуществляемые отдельно взятым потоком (произведение матрицы на вектор). CUDA использует большое число параллельно выполняемых потоков, поэтому каждому потоку ставится в соответствие один элемент матрицы или компонента вектора. Для определения номера потока ядро использует встроенные переменные threadIdx и blockIdx. Операции над векторами (скалярные произведения, умножение на скаляр, линейные комбинации) осуществляются функциями CUBLAS, соответственно cublasDdot, cublasDscal, cublasDaxpy.

Копирование данных из памяти CPU в память GPU и обратно относится к накладным вычислительным затратам и увеличивает время выполнения программы, чтобы пренебречь этими операциями, при проведении экспериментов рассматривались системы уравнений, полностью размещаемые в памяти графического ускорителя. В этом случае, в память GPU из памяти CPU копировались только матрица (в разреженном формате) и вектор правой части, а обратно — вектор решения. Вспомогательные векторы, используемые в методах подпространств Крылова формировались и хранились в памяти GPU. Кроме того учитывалась иерархия памяти графического ускорителя, так как имеется существенное отличие во времени доступа к данным, помещённым на разные уровни (глобальная память, локальная память, текстурная память, общая память, регистровая память). В таблице 1 представлены результаты применения текстурной памяти при решении систем стандартного и разрывного методов Галёркина методом сопряженных градиентов с диагональным предобуславливанием. Для хранения матриц систем уравнений использовался сжатый координатный формат (COO). Как видно из таблицы 1, для

Таблица 1. Зависимость времени решения от использования текстурной памяти

Использование текстурной памяти	МГ, $N = 2568$	СРМГ, $N = 10034$	МГ, $N = 124640$	СРМГ, $N = 496842$
Без использования	0.114	8.68	2.31	189.18
С использованием	0.116	8.44	1.88	152.99

больших систем ($\lg N > 5$) за счет применения текстурной памяти время вычислений уменьшилось примерно на 20%.

§ 4. Сравнение прямого и итерационных методов решения систем РМГ

При решении систем РМГ сравнивались как прямые методы на основе треугольных разложений (LU), так и итерационные методы подпространств Крылова (метод сопряженных градиентов (CG), метод обобщенных минимальных невязок (GMRES), метод бисопряженных градиентов (BiCGStab).

В таблицах 2 и 3 приведены численные оценки числа обусловленности матрицы A и времени вычислений для прямого (LU) и итерационного методов, полученные при решении систем классического (МГ) и симметричного разрывного метода Галёркина (СРМГ) на GPU. Для сравнения свойств систем уравнений МГ и СРМГ при проведении вычислительных экспериментов подбирались расчётные сетки с результирующими системами уравнений примерно одинакового размера. Решения, полученные прямым и итерационным методами близки друг к другу ($\|u_{LU} - u_{CG}\|_2 < 10^{-7}$).

Таблица 2. Числа обусловленности и время решения систем МГ

Размер системы N	$k_A(N)$	Время, сек LU	CG	DIAG/CG
952	2.23×10^3	0.039	0.077	0.065
2568	7.93×10^3	0.260	0.190	0.114
4960	1.41×10^4	1.063	0.236	0.188
7640	2.31×10^4	3.380	0.306	0.243
9724	2.73×10^4	—	0.347	0.282
100660	—	—	2.384	1.807

Таблица 3. Числа обусловленности и время решения систем СРМГ

Размер системы N	η	$k_A(N,\eta)$	Время, сек LU	CG	DIAG/CG
1042	5×10^3	3.17×10^7	0.042	4.72	2.71
2426	10^3	1.00×10^7	0.217	5.83	3.29
4758	10^3	1.37×10^7	0.922	8.56	4.80
7390	10^3	1.75×10^7	2.881	12.73	7.21
10034	10^3	2.14×10^7	—	15.54	8.68
100866	10^3	—	—	147.80	80.99

Благодаря хорошей обусловленности систем классического метода Галёркина итерационные методы на GPU выполняются на порядок быстрее, чем прямой метод (таблица 2, $N = 7640$). Диагональное предобуславливание сокращает время вычислений в сравнении с методом сопряженных градиентов без использования предобуславливателя примерно на 20–25%. В случае плохо обусловленных систем уравнений СРМГ существенно быстрее решаются системы прямым методом с той же точностью вычислений. Возможности диагонального предобуславливания ограничиваются только сокращением времени вычислений почти в два раза, а зависимость от шага сетки сохраняется.

Обусловленность систем уравнений СРМГ зависит не только от размера системы, но и от выбранного параметра штрафа η в (1.2). При постоянном размере системы СРМГ ($N = \text{const}$) число обусловленности увеличивается пропорционально параметру штрафа. Например, для $N = 10034$ и $\eta = 500, 1000, 5000$, число обусловленности принимает значения: $k_A(N,\eta) = 1.09 \times 10^7, 2.14 \times 10^7, 1.04 \times 10^8$. При уменьшении штрафного параметра в (1.2) требуется контроль за устойчивостью метода. Расчёты показывают, что обусловленность может быть оценена как $k_A(N,\eta) = \mathcal{O}(\eta/h)$, где $h = 1/N$ — шаг сетки. Для НРМГ с несимметричными системами уравнений число обусловленности становится ещё больше, чем для метода СРМГ.

§ 5. Предобусловленные итерационные методы

Для эффективного решения систем, подобных (1.3), на параллельных компьютерах необходимо построение предобуславливателей, обладающих достаточным ресурсом параллелизма. Отметим, что многие предобуславливатели плохо распараллеливаются как на стадии построения, так и на стадии применения в итерационном алгоритме.

В последнее время все больше внимания привлекают к себе методы предобуславливания, обладающие естественным параллелизмом, в их числе методы явного предобуславливания, для реализации которых необходимо лишь умножать предобуславливатель на вектор, что, безусловно, является более эффективной операцией при распараллеливании. Наиболее известными являются разреженные обратные предобуславливатели [13], в частности AINV, который основан на разложении обратной матрицы, выполняемой в неполном виде $M \approx A^{-1} \approx ZD^{-1}Z^T$, здесь $Z \approx L^{-T}$ из $A = LDL^T$, где L^T — нижняя треугольная матрица с единичной главной диагональю. Разложение осуществляется в ходе процесса неполной A-ортогонализации, примененной к векторам канонического базиса. Разреженность матрицы $Z = (z_{ij})$ достигается обнулением внедиагональных элементов z_{ij} меньших, чем заданное значение. Процесс построения предобуславливателей типа AINV хорошо распараллеливается.

Одним из наиболее эффективных методов решения систем большого размера, полученных при аппроксимации эллиптических и параболических задач на неструктурированных сетках, является алгебраический многосеточный метод (AMG). Алгебраический многосеточный метод также рассматривается в качестве предобуславливателя для ускорения сходимости итерационных методов решения СЛАУ. AMG работает непосредственно с решаемой системой уравнений, формируя структуру многосеточных уровней, исходя из портрета матрицы. В работе рассматривался вариант метода со сглаживанием при объеденении неизвестных. Подходы к распараллеливанию AMG для графических ускорителей представлены в [14].

Из методов предобуславливания, основанных на треугольном разложении $A = (a_{ij})$ был реализован вариант неполной LU-факторизации [13], осуществляющей приближенное разложение $A \approx LU$ с обнулением внедиагональных малых элементов верхней U и нижней L треугольных матриц в виде ILU(ρ, τ), где ρ — максимальное число внедиагональных элементов в неполном LU-разложении, τ — нижняя граница для внедиагональных элементов предобуславливателя, элементы меньшие τ полагаются равными 0.

В работе использовался также диагональный предобуславливатель $M = (a_{ii}^{-1})$ (далее DIAG), который обеспечивает меньшее ускорение сходимости, чем приведенные выше подходы, но, вместе с тем, требует меньших затрат на построение, хранение предобуславливателя, и на его применение на каждой итерации метода. Кроме того, он прост в распараллеливании и поддерживает мелкозернистый параллелизм, что существенно при вычислениях на GPU.

§ 6. Результаты решения СЛАУ на графических процессорах

Вычислительные эксперименты были проведены на системе, оснащенной графическим ускорителем NVIDIA GeForce GTX 470 и двухядерным процессором AMD Athlon 64 X2 5600+ с тактовой частотой 1 ГГц, кэш-памятью второго уровня 2 МБ и оперативной памятью 2 ГБ. Видеокарта GeForce GTX 470 обладает большим числом ядер (448) и объёмом собственной памяти (1280 МБ), но ее потоковые процессоры уступают в тактовой частоте процессору AMD, поэтому ускорение вычислений достигается за счет запуска большого числа потоков (1024 потока в блоке). GeForce GTX 470 соответствует CUDA Compute capability 2.0, что позволило выполнить все эксперименты с двойной точностью вычислений.

При решении симметричных систем классического метода Галёркина (МГ) и симметричного разрывного метода Галёркина (СРМГ) сравнивались время построения предобуславливателя и время работы итерационного алгоритма на CPU и GPU (см. таблицу 5). Вариант CPU соответствует выполнению программы на одном ядре процессора. Условием выхода из итерационного процесса для всех рассмотренных алгоритмов решения СЛАУ было $\|f - Au_i\|_2 < 10^{-8}\|f\|_2$.

Таблица 4. Время построения предобуславливателя/время решения системы и (число итераций) для МГ и СРМГ

Метод	МГ $N = 2568$		СРМГ $N = 10034$		МГ $N = 124640$		СРМГ $N = 496842$
	CPU	GPU	CPU	GPU	CPU	GPU	GPU
CG	0.17 (205)	0.19 (205)	76.77 (16468)	15.54 (16468)	92.15 (1487)	3.23 (1483)	371.31 (51410)
DIAG/CG		0.11 (159)		8.68 (9190)		2.31 (1115)	189.18 (26280)
AINV/CG		0.04/0.10 (96)		0.41/7.49 (5690)		2.35/2.53 (726)	20.95/192.74 (14981)
AMG/CG		0.04/0.08 (112)		0.06/3.98 (5318)		0.23/2.76 (615)	0.87/207.04 (12541)
ILU(p,τ)/CG		$(30, 10^{-5})$ 0.97/0.25 (22)		$(60, 10^{-6})$ 11.06/1.77 (20)		$(200, 10^{-3})$ 59.23/75.33 (76)	—

Варианты МГ и СРМГ были получены на одной и той же расчётной сетке, но соответствующие им системы состояли из $N = 2568$ и $N = 10034$ уравнений. Также проводились исследования для систем РМГ большего размера ($N = 124640$ — для МГ и $N = 496842$ — для СРМГ) построенных на сетке, состоявшей из 62911 узлов и 62400 билинейных конечных элементов. Ускорение параллельных вычислений на GPU при решении системы уравнений СРМГ ($N = 10034$) методом сопряженных градиентов (CG) составило почти пять раз. При увеличении размера систем достигнуто ускорение 28.5 раз для системы МГ (при $N = 124640$) и примерно 35 раз для системы СРМГ (при $N = 496842$ система решалась на CPU методом сопряженных градиентов 12920.1 сек.). Использование DIAG/CG в случае стандартного метода Галёркина уменьшило время вычислений в 1.3–1.4 раза. В случае СРМГ, время решения сократилось в 1.8–2 раза. Применение предобуславливателя AINV для больших систем оказалось менее эффективным, чем диагональное предобуславливание, но тем не менее обеспечило ускорение в 1.7 раза в сравнении с CG.

В данной работе также использовался трехуровневый алгебраический AMG предобуславливатель. Степень распараллеливания метода AMG/CG оказалась достаточно высокой и для небольших систем метод оказался более эффективным, чем диагональный предобуславливатель. Затраты на построение предобуславливателя небольшие, но суммарные затраты на решение систем (при $\lg N > 5$) уже больше, чем для диагонального предобуславливания. Несмотря на это применение AMG предобуславливания позволило сократить время вычислений в 1.8 раза. Для больших систем время решения на графическом устройстве с применением AMG/CG примерно такое же как у AINV/CG и больше, чем у DIAG/CG на 10%.

Построение предобуславливателя со сходимостью за минимальное число итераций обеспечил метод ILU (ρ, τ) факторизации, которая выполнялась последовательно на CPU. Однако ILU применялся как неявный предобуславливатель, что связано с увеличением вычислительных затрат по сравнению с другими алгоритмами. Следует отметить необходимость подбора оптимальных параметров ρ и τ для каждой решаемой системы ($(30, 10^{-5})$ — для СМГ и $(60, 10^{-6})$ — для СРМГ в случае небольших систем и $(200, 10^{-3})$ — для системы $N = 124640$).

В случае несимметричных систем уравнений (НРМГ) алгоритмы GMRES, BiCGStab без предобуславливания сходятся очень медленно (число итераций более чем на порядок превышает размер системы) или не сходятся вообще. Наиболее эффективным оказалось применение предобуславливателя ILU(ρ, τ). Для системы НРМГ ($N = 7390$), решаемой на GPU, в таблице 5 приведены время построения предобуславливателя, время решения системы и число итераций. Вычислительные затраты были минимальны в случае выбора параметров $(30, 10^{-5})$.

В этом случае время решения для GMRES и BiCGStab примерно одинаково. Данная система решается на GPU прямым методом в два раза быстрее, чем итерационными методами.

Таблица 5. ILU-предобуславливание системы НРМГ, решаемой на GPU

Предобуславливатель	Метод решения	
	GMRES	BiCGStab
$\mathrm{ILU}(30, 10^{-6})$	2.13/5.79 (120)	2.13/6.45 (95)
$\mathrm{ILU}(30, 10^{-5})$	3.11/3.26 (90)	3.11/3.03 (48)

При решении симметричных и несимметричных систем РМГ небольшого размера ($\lg N < 5$), на графическом ускорителе GeForce GTX 470 более эффективным оказался прямой метод. С увеличением размера решаемых систем — предобусловленные итерационные методы. Тем не менее, треугольные разложения A оправданы в параллельных алгоритмах метода дополнения Шура при решении больших систем.

Для достижения максимального ускорения на GPU рассматривалось несколько форматов сжатого хранения матрицы коэффициентов $A = \{a_{ij}\}$ из (1.4): координатный формат COO, в котором хранятся упорядоченные по строкам значения элементов $a_{ij} \neq 0$ и их строчные i и столбцовые индексы j; разрежённый строчный формат CSR (вместо строчных индексов хранятся указатели на начальные позиции каждой строки в списках $a_{ij} \neq 0$ и j); BCSR2 — блочный вариант CSR, использующий блоки 2×2; формат ELL — с постоянным числом элементов в строке (строки дополняются до ширины «ленты» нулевыми элементами) и формат HYB — комбинация ELL и формата COO.

Таблица 6. Время решения DIAG/CG для различных форматов хранения матриц

Формат хранения матрицы	СМГ, $N = 2568$		СРМГ, $N = 10034$		СМГ, $N = 124640$		СРМГ, $N = 496842$	
	CPU	GPU	CPU	GPU	CPU	GPU	CPU	GPU
CSR	0.15	0.03	43.19	7.84	70.58	5.27	6767.74	960.69
BCSR2		0.07		3.57		6.58		413.60
COO		0.11		8.68		2.31		189.18
HYB		0.05		1.69		1.31		95.69
ELL		0.02		1.47		1.17		88.23

Полученные при решении системы СРМГ результаты показывают (таблица 6), что наиболее эффективным оказался формат ELL, в котором доступ к памяти непрерывный. В свою очередь, недостаточная эффективность формата CSR при вычислениях на GPU связана с «разрывным» доступом к памяти. Это согласуется с тем, что структура рассматриваемых систем уравнений сочетала свойства как структурированных, так и неструктурированных сеток.

Проведенные сравнения показали эффективность применения видеокарты в качестве ускорителя вычислений при решении СЛАУ. Достигнутое на больших системах ускорение соответствует применению тридцати процессорных ядер, аналогичных использованному в работе CPU, при 100% эффективности распараллеливания алгоритмов. Вместе с тем, характеристика «цена —производительность» для такой вычислительной системы будет на порядок меньше, чем для системы с графическим ускорителем GeForce GTX 470.

Следует отметить, что для эффективного решения систем РМГ на графических ускорителях требуются предобуславливатели с хорошими аппроксимационными свойствами шаблона метода Галёркина с разрывными функциями, учитывающие вклады внутренних и граничных аппроксимаций, параллельные по построению и явные по применению, а также необходимо учитывать особенности архитектуры GPU при хранении матриц и предобуславливателей.

При решении больших систем уравнений РМГ ($\lg N > 6$) необходимо распараллеливание для multiGPU средствами CUDA на одном вычислительном узле и MPI+CUDA — на нескольких, и применение гибридных методов решения.

СПИСОК ЛИТЕРАТУРЫ

1. Arnold D. N., Brezzi F., Cockburn B., Marini L. D. Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems // SIAM J. Numerical Analysis. 2002. Vol. 39. P. 1749–1779.
2. Klockner A., Warburton T., Bridge J., Hesthaven J.S. Nodal Discontinuous Galerkin Methods on Graphics Processors // Journal of Computational Physics. 2009. Vol. 228. № 21. P. 7863–7882.
3. Cabel T., Charles J., Lanteri S. Multi-GPU acceleration of a DGTD method for modeling human exposure to electromagnetic waves // Research Report № RR-7592. INRIA Sophia Antipolis — CNRS. 2011.
4. Cecka C., Lew A., Darve E. Introduction to assembly of finite element methods on graphics processors // IOP Conf. Ser.: Mater. Sci. Eng. 2010. Vol. 10. № 1.
5. Markall G. R., Ham D. A., Kelly P.H.J. Towards generating optimised finite element solvers for GPUs from high-level specifications // Procedia Computer Science. 2010. Vol. 1. № 1. P. 1815–1823.
6. Боресков А. В., Харламов А. А. Основы работы с технологией CUDA. М.: ДМК Пресс, 2010. 232 с.
7. Saad Y. Iterative Methods for Sparse Linear Systems. SIAM, 2000. 439 p.
8. Копысов С. П., Сагдеева Ю. А. Разрывные методы Галёркина для задач теории упругости // Сб. статей конференции «Актуальные проблемы математики, механики, информатики». Екатеринбург: ИММ УрО РАН, 2009. С. 72–78.
9. Castillo P. Performance of discontinuous Galerkin methods for elliptic PDE's // SIAM J. Scientific Computing. 2002. Vol. 24. № 2. P. 524–547.
10. CUDA Toolkit 4.0 CUBLAS Library `http://developer.download.nvidia.com/compute/cuda/4_0_rc2/toolkit/docs`
11. Копысов С. П., Новиков А. К. Метод декомпозиции для параллельного адаптивного конечно-элементного метода // Вестник Удмуртского университета. Математика. Механика. Компьютерные науки. 2010. Вып. 3. С. 141–152.
12. Копысов С. П., Красноперов И. В., Рычков В. Н. Объектно-ориентированный метод декомпозиции области // Вычислительные методы и программирование. 2003. Т. 4. № 1. С. 176–193.
13. Chen K. Matrix Preconditioning Techniques and Applications. Cambridge University Press, 2005. 601 p.
14. Haase G., Liebmann M., Douglas C., Plank G. A Parallel Algebraic Multigrid Solver on Graphics Processing Units // Lecture Notes in Computer Science. 2010. Vol. 5938. P. 38–47.

Поступила в редакцию 05.05.11

S. P. Kopysov, A. K. Novikov, Yu. A. Sagdeeva
Solving of discontinuous Galerkin method systems on GPU

Solving systems of equations obtained in Discontinuous Galerkin method by GPU-computing is considered. The direct method and iterative Krylov methods with preconditioning are used. The performance of GPU-computing for these systems of equations is compared with one of multicore CPU.

Keywords: discontinuous Galerkin method, system of linear algebraic equations, Krylov subspaces methods, preconditioner, sparse matrices, general purpose computing on GPU.

Mathematical Subject Classifications: 65Y05, 65N30, 65N22, 65F08, 65F50

Копысов Сергей Петрович, д. ф.-м. н., профессор, кафедра вычислительной механики, Удмуртский государственный университет, 426034, Россия, г. Ижевск, ул. Университетская, 1;
заведующий лабораторией, Институт прикладной механики УрО РАН, 426067, Россия, г. Ижевск, ул. Т. Барамзиной, 34.
E-mail: s.kopysov@gmail.com

Новиков Александр Константинович, к. ф.-м. н., доцент, кафедра вычислительной механики, Удмуртский государственный университет, 426034, Россия, г. Ижевск, ул. Университетская, 1;
старший научный сотрудник, Институт прикладной механики УрО РАН, 426067, Россия, г. Ижевск,

ул. Т. Барамзиной, 34.
E-mail: an@udman.ru

Сагдеева Юлия Альбертовна, к. ф.-м. н., доцент, кафедра вычислительной механики, Удмуртский государственный университет, 426034, Россия, г. Ижевск, ул. Университетская, 1;
старший научный сотрудник, Институт прикладной механики УрО РАН, 426067, Россия, г. Ижевск, ул. Т. Барамзиной, 34.
E-mail: sagdeeva@yandex.ru

Kopysov Sergei Petrovich, Doctor of Physics and Mathematics, Professor, Department of Computational Mechanics, Udmurt State University, ul. Universitetskaya, 1, Izhevsk, 426034, Russia
Head of the Laboratory, Institute of Applied Mechanics, Ural Branch of RAS, ul. T. Baramzinoi, 34, Izhevsk, 426067, Russia

Novikov Aleksandr Konstantinovich, Candidate of Physics and Mathematics, Associate Professor, Department of Computational Mechanics, Udmurt State University, ul. Universitetskaya, 1, Izhevsk, 426034, Russia
Senior Researcher, Institute of Applied Mechanics, Ural Branch of RAS, ul. T. Baramzinoi, 34, Izhevsk, 426067, Russia

Sagdeeva Yuliya Al'bertovna, Candidate of Physics and Mathematics, Associate Professor, Department of Computational Mechanics, Udmurt State University, ul. Universitetskaya, 1, Izhevsk, 426034, Russia
Senior Researcher, Institute of Applied Mechanics, Ural Branch of RAS, ul. T. Baramzinoi, 34, Izhevsk, 426067, Russia