УДК 532.5.032+519.6

АНАЛИЗ ПРОИЗВОДИТЕЛЬНОСТИ ГИДРОДИНАМИЧЕСКИХ РАСЧЕТОВ НА GPU- И CPU-КЛАСТЕРАХ

А. В. Сентябов[1], А. А. Гаврилов[2], М. А. Кривов[3], А. А. Дектерев[4], М. Н. Притула[5]

Рассматривается ускорение параллельных гидродинамических расчетов на кластерах с CPU- и GPU-узлами. Для тестирования используется собственный CFD-код SigmaFlow, портированный для расчетов на графических ускорителях с помощью технологии CUDA. Алгоритм моделирования течения несжимаемой жидкости основан на SIMPLE-подобной процедуре и дискретизации с помощью метода контрольного объема на неструктурированных сетках из гексаэдральных ячеек. Сравнение скорости расчета показывает высокую производительность графических ускорителей нового поколения в GPGPU-расчетах.

Ключевые слова: GPGPU, численное моделирование, вычислительная гидродинамика, SIMPLE, MPI, CUDA.

1. Введение. В настоящее время графические ускорители прочно вошли в мир высокопроизводительных вычислений, приведя к появлению широкого класса технологий GPGPU (General-Purpose computing for Graphics Processing Units). Привлекательность графических ускорителей (GPU, Graphics Processing Unit) с точки зрения вычислительной производительности привела к их широкому применению в суперкомпьютерных системах. В 47-й редакции top-500 мощнейших компьютеров мира (июнь 2016 г., [1, 2]) 67 систем содержат видеокарты NVidia, в частности в России GPU используются в кластерах “Ломоносов”, “Ломоносов-2” [3] и др. Тематика вычислений, выполняемых на GPU, довольно широка и включает в себя молекулярную динамику [4], квантовую механику [5], химию [6], газовую динамику [7] и многое другое. Как правило, в этих областях используются алгоритмы, легко поддающиеся распараллеливанию на огромное число относительно независимых потоков, что идеально подходит для реализации на графических ускорителях. Пиковая производительность GPU быстро растет и многократно превышает таковую у центральных процессоров [8]. Тем не менее, одна из самых ресурсоемких областей моделирования — вычислительная гидродинамика (CFD, Computational Fluid Dynamics) — представлена в GPGPU-вычислениях не так широко, что связано как с трудностями реализации алгоритмов, так и с ограничениями графических ускорителей.

Большинство алгоритмов вычислительной гидродинамики несжимаемой жидкости основано на решении уравнения эллиптического типа для поправки давления, которое связывает всю область течения в каждый момент времени. В итоге, реализация вычислений на GPU приводит к необходимости переписывания алгоритмов всех основных ресурсоемких операций для распараллеливания на ядрах GPU. При этом в памяти GPU требуется хранить все необходимые для расчета данные, поскольку их пересылка между GPU и CPU (Central Processing Unit) занимает очень много времени. Учитывая ограниченный объем памяти графических ускорителей, это налагает серьезные ограничения на решаемые на GPU задачи. Эта трудность устраняется при использовании нескольких GPU, что приводит к появлению еще одного уровня параллелизма. В то же время известно, что увеличение количества вычислительных узлов для решения задачи, в конце концов, приводит к насыщению, при котором рост производительности практически не наблюдается. Это происходит из-за роста расходов на обмен информацией между вычислительными узлами. При этом чем больше размер расчетной сетки, тем большее количество узлов можно эффективно задействовать для расчета на ней. В подавляющем большинстве случаев при использовании

[1] Институт теплофизики им. С. С. Кутателадзе СО РАН (ИТ СО РАН), просп. Лаврентьева, 1, 630090, г. Новосибирск; мл. науч. сотр., e-mail: sentyabov_a_v@mail.ru

[2] Институт теплофизики им. С.С. Кутателадзе СО РАН (ИТ СО РАН), просп. Лаврентьева, 1, 630090, г. Новосибирск; вед. инженер, e-mail: gavand@yandex.ru

[3] ООО “ТТГ Лабс”, ул. Нобеля, д. 7, ИЦ “Сколково”, 143026, Москва; генеральный директор, e-mail: m_krivov@ttgLabs.com

[4] Институт теплофизики им. С.С. Кутателадзе СО РАН (ИТ СО РАН), просп. Лаврентьева, 1, 630090, г. Новосибирск; ст. науч. сотр., e-mail: dekterev@mail.ru

[5] ООО “ТТГ Лабс”, ул. Нобеля, д. 7, ИЦ “Сколково”, 143026, Москва; разработчик, e-mail: m_pritula@ttgLabs.com

GPU обмен информацией между вычислительными узлами ограничивается также и скоростью пересылки информации между GPU и CPU по шине PCI-E (PCI-Express).

Для выявления указанных ограничений и сравнительного анализа различных вычислительных узлов на суперкомпьютерах ниже проводится исследование скорости параллельных гидродинамических расчетов на двух типах CPU-узлов и двух типах GPU-узлов кластеров "Ломоносов" и "Ломоносов-2".

2. Математическая модель и программная реализация.

2.1. Основные уравнения и методы решения. Реализация GPGPU-вычислений проводилась на основе программного комплекса SigmaFlow [9, 10]. Рассматривалось трехмерное вязкое нестационарное несжимаемое течение жидкости, которое описывается уравнениями Навье–Стокса:

$$\nabla \cdot \boldsymbol{v} = 0, \quad \frac{\partial(\rho \boldsymbol{v})}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v}) = -\nabla p + \nabla \cdot \boldsymbol{T},$$

где $\boldsymbol{T}_{ij} = \mu\left(\dfrac{\partial v_i}{\partial x_j} + \dfrac{\partial v_j}{\partial x_i}\right)$ — тензор вязких напряжений, v — скорость, ρ — плотность, t — время, p — давление, x — координата, μ — динамическая вязкость.

Используемый в нашей работе численный алгоритм базируется на методе конечного объема для неструктурированной сетки. Детали его реализации для расчета ламинарных течений подробно описаны в статьях [9, 10]. Для построения разностного аналога дифференциального уравнения второго порядка в методе конечного объема используются сеточные распределения искомого поля и его градиента. Значение градиента поля в центре контрольного объема рассчитывается явным способом по значениям поля в центрах контрольных объемов с использованием метода наименьших квадратов [11].

Одна из основных трудностей при разработке численных алгоритмов на базе сеточных методов для уравнений Навье–Стокса — это определение связи поля давления с полем скорости. В данной работе связь между полями скорости и давления, обеспечивающая выполнение уравнения неразрывности, реализуется при помощи SIMPLE-подобной процедуры на совмещенных сетках. В литературе можно встретить большое количество вариантов методики расщепления для уравнений гидродинамики и способов вывода результирующих выражений. Дополнительную информацию по сравнению и использованию SIMPLE- и SIMPLEC-методов можно найти в [12–15].

Значения полей скорости и давления хранятся в одних и тех же узлах (совмещенные сетки). Подход, при котором значения полей скорости и давления расположены в одних и тех же узлах (центрах контрольных объемов), наиболее экономичен с точки зрения программной реализации. Для устранения возможных шахматных осцилляций решения используется подход Рхи–Чоу, предложенный в работе [16]. Аппроксимация конвективных членов уравнения переноса количества движения осуществлялась с помощью противопоточной схемы QUICK (Quadratic Upwind Interpolation for Convective Kinematics) [17]. Вязкие слагаемые аппроксимировались со вторым порядком точности. Для интегрирования по времени использовалась неявная направленная трехслойная схема второго порядка точности. Системы разностных уравнений, аппроксимирующие уравнение движения, решались итерационным методом неполной LU-факторизации [18]. Система линейных уравнений, полученная в результате дискретизации эллиптического уравнения на поправку давления, решалась с помощью трехслойного вариационного метода сопряженных невязок [19].

2.2. Технологии распараллеливания и расчетов на GPU. Для ускорения расчетов применялась технология параллельных вычислений, основанная на декомпозиции расчетной области. Декомпозиция области решения заключается в разбиении ее на неперекрывающиеся (соприкасающиеся) односвязные подобласти. При геометрической декомпозиции расчетной области для использования параллельных вычислений в массивы данных включается информация о данных из соседних подобластей. В качестве коммуникационного интерфейса применялся протокол MPI. Для получения хорошей сбалансированности процессоров необходимо каждому процессору выделить примерно одинаковую часть работы. Распределение расчетных узлов по процессорам осуществляется с помощью программы MeTiS [20].

В GPU-версии кода все основные операции по обработке расчетных данных выполнялись на графическом процессоре: расчет градиентов полей скорости и давления, дискретизация уравнения движения и уравнения на поправку давления, решение систем линейных уравнений и корректирующие операции процедуры SIMPLE [21]. Для их реализации использовалась архитектура CUDA (Compute Unified Device Architecture) поколения 2.0. В подавляющем большинстве реализованных CUDA-ядер требуется выполнить некую операцию над каждым узлом или гранью сетки, оперируя значениями с соседних с ним/ней узлов. Таким образом, структуру подобных ядер можно описать следующим образом. Имеется массив размера, равного количеству контрольных объемов или граней сетки. Для каждого элемента необходимо прочитать значения из этого же массива по известным индексам и подать их на вход некой функции,

которая выполняется в отдельной нити CUDA на GPU-ядре. Так же, как и в CPU-версии, использовалась декомпозиция расчетной области: в этом случае роль ядра CPU, на котором выполняется отдельный поток, играет весь графический процессор. Таким образом, осуществляется разбиение задачи между несколькими GPU.

Для тестирования использовались два кластера, содержащие CPU- и GPU-узлы, характеристики которых приведены в таблице.

Характеристики вычислительных узлов

Вычислительный узел	Частота, ГГц	Объем памяти узла, Гб
Nvidia Tesla C2070	1,15	6
Nvidia Tesla K40	0,745	12
Intel Xeon 5670	2,93	48
Intel Xeon E5-2697v3	2,6	64

Рис. 1. Обтекание цилиндра: геометрия задачи

2.3. Тестовое течение. В качестве тестового течения рассмотрено нестационарное пространственное ламинарное обтекание однородным потоком жидкости круглого цилиндра. Число Рейнольдса определяется по диаметру цилиндра и скорости набегающего потока: $\mathrm{Re} = \rho U_{\mathrm{in}} D/\mu$, где U_{in} — скорость вдали от цилиндра, D — диаметр цилиндра. При числе Рейнольдса $\mathrm{Re} = 100$, рассмотренном в данном случае, происходит периодический отрыв вихрей, образующих дорожку Кармана в следе за цилиндром.

Геометрия расчетной области представлена на рис. 1. Размер внешней границы $D_{\mathrm{ext}} = 40D$. Длина цилиндра составляла $4D$. На входной границе области задавались параметры равномерного потока. На выходной границе ставились "неотражающие" граничные условия. На торцевых границах задавались условия симметрии.

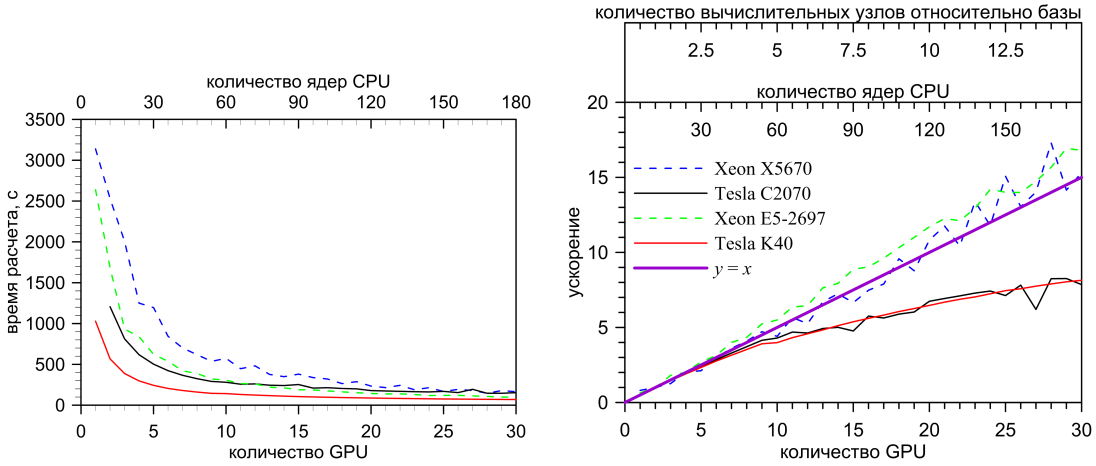
Сетки O-типа содержали 500 узлов вдоль радиуса и 1000 узлов по окружности со сгущениями к обтекаемому цилиндру и к области следа за цилиндром. Вдоль цилиндра сетки содержали 10, 20, 40 и 100 узлов и соответственно 5, 10, 20 и 50 миллионов ячеек, заполняющих трехмерную область.

Нестационарный расчет проводился с начального приближения, равного нулю, с шагом по времени $\tau = 0.04\,T_{\mathrm{ref}}$, где $T_{\mathrm{ref}} = D/U_{\mathrm{in}}$ — характерное время течения. Расчет выполнялся в течении $0.6\,T_{\mathrm{ref}}$ (15 временны́х шагов). На каждый временной слой приходилось 30 итераций метода SIMPLEC.

3. Результаты расчетов. Гидродинамические расчеты могут выполняться как с одинарной (float), так и с двойной (double) точностью представления вещественных чисел. Большинство расчетов не требует двойной точности, однако в некоторых задачах одинарной точности оказывается недостаточно. Для программного кода, исполняемого на CPU, использование двойной точности не представляет проблем. При использовании графических ускорителей ситуация радикально меняется. Во-первых, производительность GPU при расчетах с двойной точностью гораздо ниже, чем с одинарной. Во-вторых, не все возможности технологии CUDA доступны для операций с числами типа double. В-третьих, становится сильнее ограничение по памяти GPU.

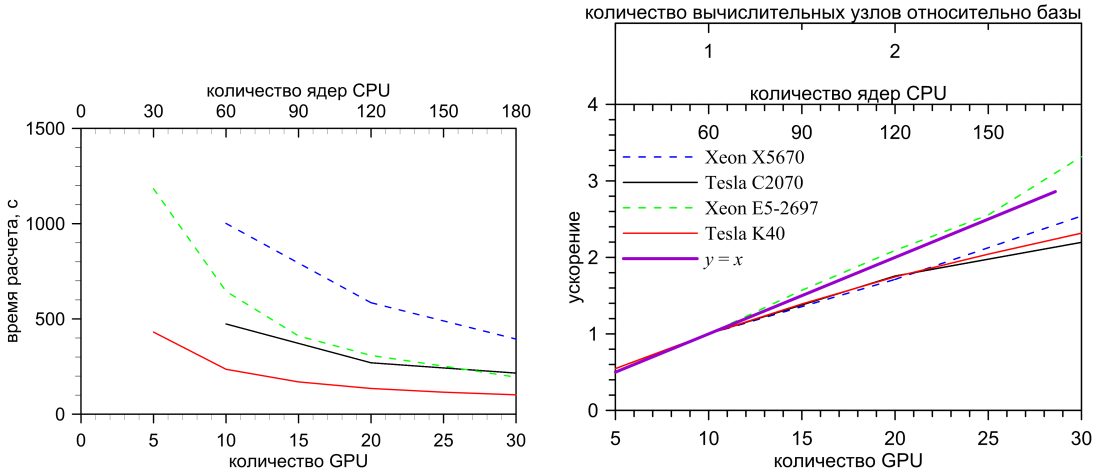
Расчеты были проведены на различных сетках с использованием как одинарной, так и двойной точности (CPU и GPU). На сетке, содержащей 5 миллионов ячеек, были проведены расчеты с использованием от 6 до 180 ядер CPU с шагом 6 ядер. Для GPGPU-расчетов нижняя граница количества GPU определялась требуемым объемом памяти, а верхняя составляла 30 GPU. Для детальных сеток было рассмотрено только несколько вариантов разбиений: с шагом 5 узлов для GPU Tesla K40, 10 узлов — для GPU Tesla C2070, с шагом 30 ядер для CPU Xeon E5-2697v3 и 60 ядер — для Xeon 5670.

3.1. Расчеты с одинарной точностью. На рис. 2а–5а приведено время, затрачиваемое ЭВМ на выполнение расчета течения. На рис. 2б–5б приведено ускорение параллельных расчетов, т.е. отношение времени расчета на базовом количестве вычислительных узлов ко времени расчета на заданном количестве вычислительных узлов. При расчетах на сетке, содержащей 5 миллионов ячеек, расчеты на GPU



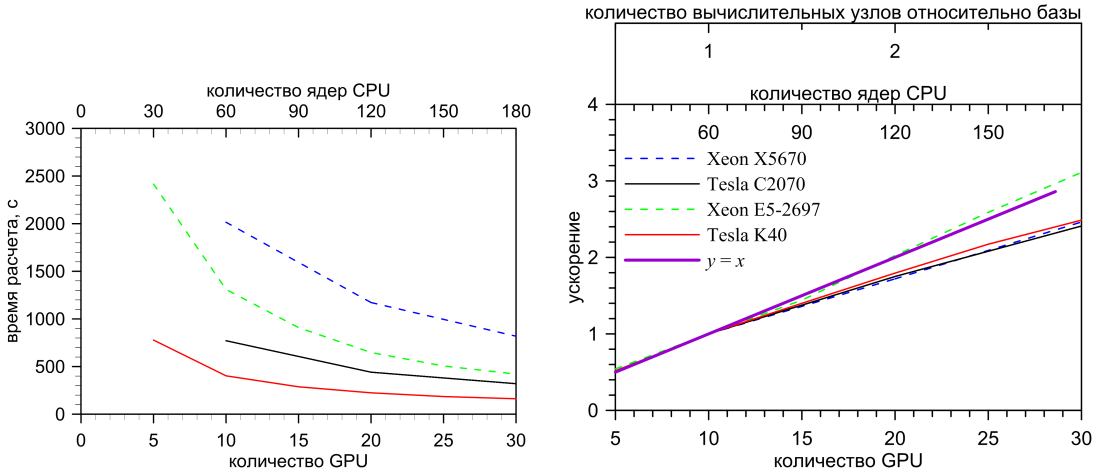
а) б)

Рис. 2. Время (а) и ускорение (б) расчетов с одинарной точностью.
Базовое количество узлов: 2 GPU и 12 ядер CPU. Сетка 5 млн ячеек



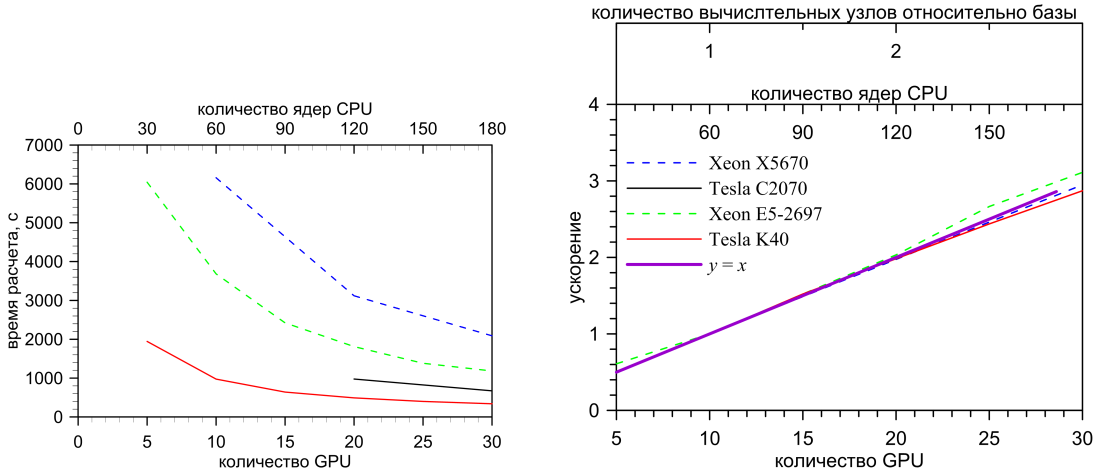
а) б)

Рис. 3. Время (а) и ускорение (б) расчетов с одинарной точностью.
Базовое количество узлов: 10 GPU и 60 ядер CPU. Сетка 10 млн ячеек



а) б)

Рис. 4. Время (а) и ускорение (б) расчетов с одинарной точностью.
Базовое количество узлов: 10 GPU и 60 ядер CPU. Сетка 20 млн ячеек

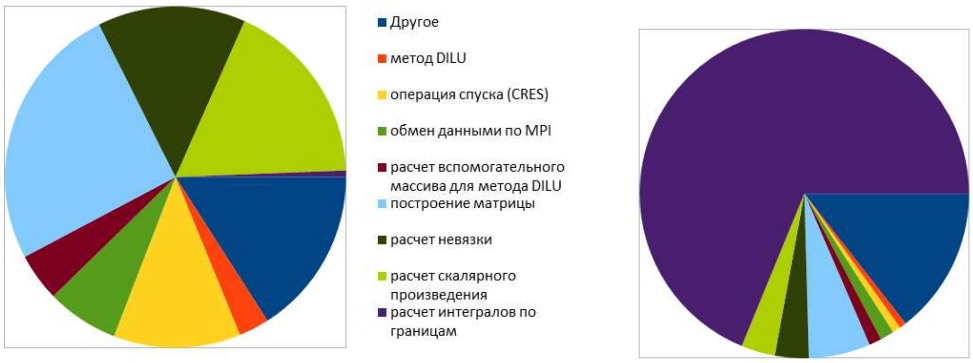


а) б)

Рис. 5. Время (а) и ускорение (б) расчетов с одинарной точностью.
Базовое количество узлов: 10 GPU и 60 ядер CPU. Сетка 50 млн ячеек

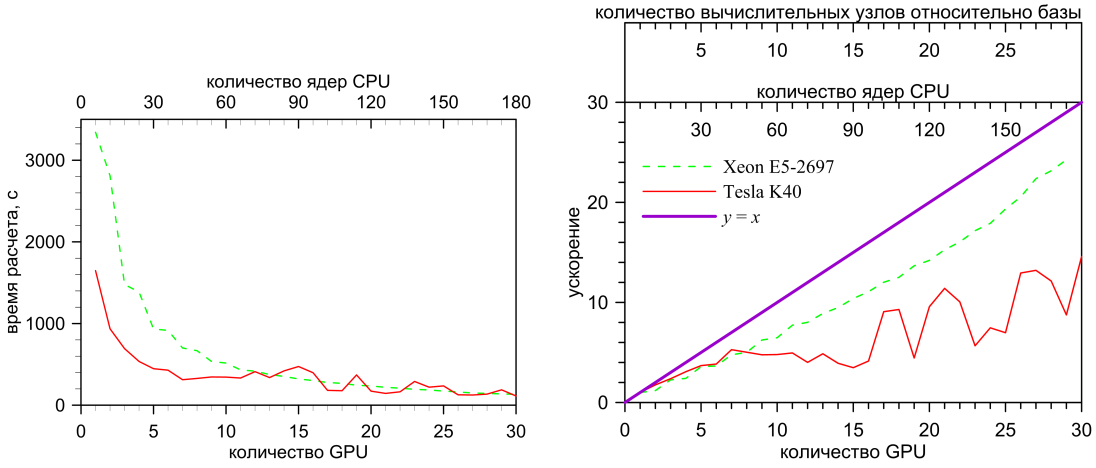
показали гораздо более высокую производительность по сравнению с CPU-узлами, если число используемых узлов не велико (рис. 2а). GPU Tesla K40 быстрее 6-ядерного процессора Xeon 5670 в 3.1 раза и быстрее 6 ядер Xeon E5-2697v3 в 2.6 раз (рис. 2). Скорость расчета на одной GPU Tesla K40 соответствует скорости расчета на 18 ядрах Xeon E5-2697v3 или на 30 ядрах Xeon 5670. Примерно для 10 GPU происходит насыщение и производительность слабо растет при увеличении количества вычислительных узлов, а эффективность распараллеливания падает (рис. 2б). При максимальном количестве узлов (30 GPU) Tesla C2070 почти не отличаются по производительности от 180 ядер Xeon 5670 и уступают 180 ядрам Xeon E5-2697v3 в 1.5 раз. В то же время, 30 GPU Tesla K40 в 1.4 раза быстрее, чем 180 ядер Xeon E5-2697.

При увеличении размера сетки различия между производительностью вычислительных узлов проявляется отчетливее, а эффективность распараллеливания на GPU увеличивается (рис. 3–5). Так, на сетке, содержащей 10 миллионов ячеек, 5 GPU Tesla K40 в 2.7 раза быстрее, чем 30 ядер Xeon E5-2697v3, а на сетке, содержащей 50 миллионов ячеек, — в 3.1 раза. Чем детальнее сетка, тем больше преимущество в производительности GPU-узлов, однако сильнее становится ограничение по памяти для GPU Tesla C2070. Новые графические ускорители Tesla K40, обладающие 12 Гб памяти, позволяют на пяти GPU-узлах проводить расчет на сетке, содержащей 50 миллионов ячеек. Ускорение параллельных расчетов на CPU Xeon E5-2697v3 близко или даже немного лучше линейного для всех расчетных сеток. Ускорение параллельных расчетов на GPU становится все ближе к линейному с увеличением количества узлов расчетной сетки. Так, для сетки, содержащей 50 миллионов ячеек, ускорение мало отличается от линейного для всех кластеров.



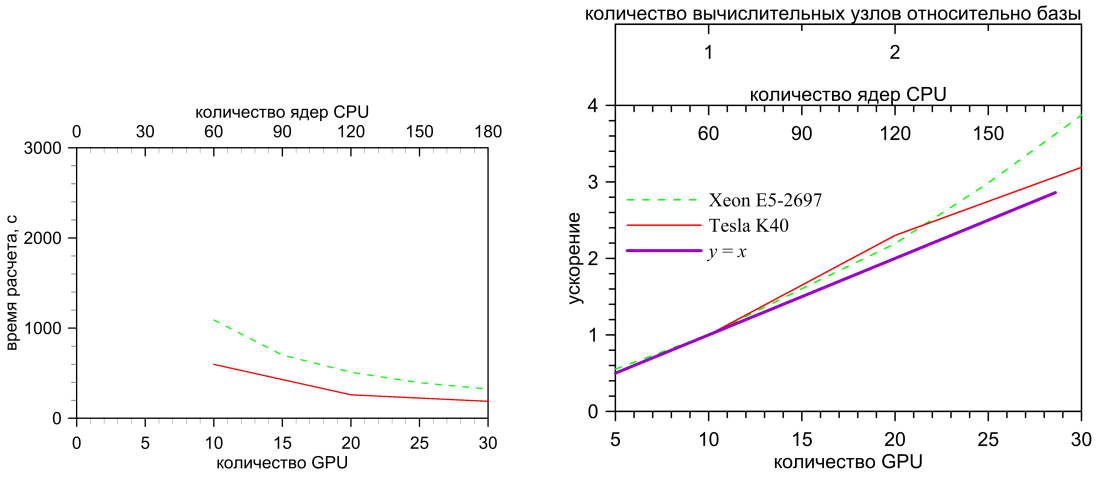
а) б)

Рис. 6. Доля некоторых операций в общем времени расчета на GPU
для одинарной (а) и двойной (б) точности



а) б)

Рис. 7. Время (а) и ускорение (б) расчетов с двойной точностью.
Базовое количество узлов: 1 GPU и 6 ядер CPU. Сетка 5 млн ячеек



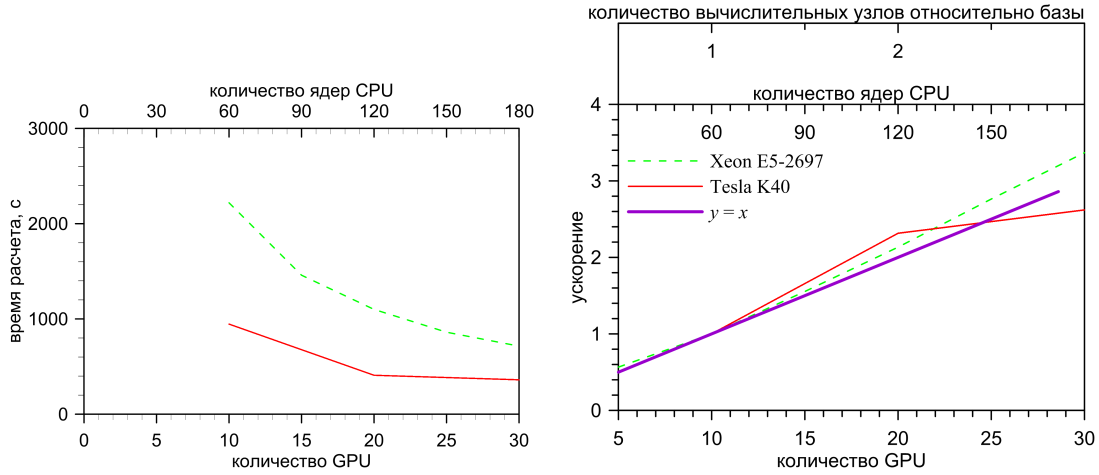
а) б)

Рис. 8. Время (а) и ускорение (б) расчетов с двойной точностью.
Базовое количество узлов: 10 GPU и 60 ядер CPU. Сетка 10 млн ячеек

3.2. Расчеты с двойной точностью. Расчет на GPU с двойной точностью представления вещественных чисел является отдельной актуальной задачей. Как показывают результаты профилирования, в этом случае скорость выполнения отдельных операций резко падает. Как видно из рис. 6, бóльшую часть времени занимает расчет интегралов по границам расчетной области. Эта процедура использует операцию суммирования элементов массива, которая в случае одинарной точности эффективно выполняется с помощью атомарных операций CUDA. Поскольку операция расчета интегральных потоков через границы не обязательна для расчета в данной задаче, дальнейшие результаты приведены без нее.

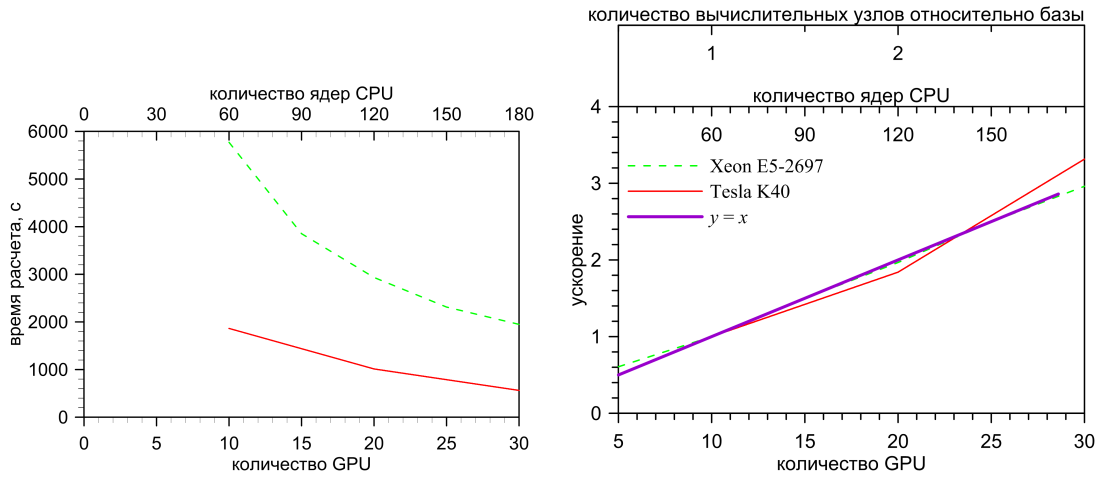
Производительность GPU Tesla K40 в 2.0 раз выше, чем 6 ядер CPU Xeon E5-2697 при расчете на сетке, содержащей 5 миллионов ячеек (рис. 7а). В то же время насыщение наступает уже при семи GPU, после чего время расчета уже не сокращается, а ускорение при распараллеливании задачи не растет (рис. 7б). Для CPU увеличение количества задействованных узлов вплоть до 180 ядер приводит к уменьшению времени расчета, которое становится примерно одинаковым со временем расчета на GPU. Для больших сеток GPU также производительнее, чем CPU, а насыщение наступает позже. Так, 30 GPU быстрее, чем 180 ядер CPU в 1.7 раза для сетки, содержащей 10 миллионов ячеек, в 2.0 раза для сетки, содержащей 20 миллионов ячеек, и в 3.5 раз для сетки, содержащей 50 миллионов ячеек (рис. 8а–10а). Ускорение при распараллеливании для больших сеток (10–20 миллионов ячеек) близко к линейному как для CPU Xeon E5-2697, так и для GPU Tesla K40 (рис. 8б–10б).

4. Заключение. Численные расчеты показывают заметный прирост производительности как GPU-, так и CPU-узлов при переходе к новым поколениям процессоров (Tesla K40 и Xeon E5-2697v3 соответ-



а) б)

Рис. 9. Время (а) и ускорение (б) расчетов с двойной точностью.
Базовое количество узлов: 10 GPU и 60 ядер CPU. Сетка 20 млн ячеек



а) б)

Рис. 10. Время (а) и ускорение (б) расчетов с двойной точностью.
Базовое количество узлов: 10 GPU и 60 ядер CPU. Сетка 50 млн ячеек

ственно). GPU демонстрируют более высокую производительность по сравнению с центральными процессорами. На сетке, содержащей 5 миллионов ячеек, GPU Tesla K40 выполняет расчет примерно с той же скоростью, что и на 18 ядрах Xeon E5-2697v3 или на 30 ядрах Xeon 5670, причем с ростом расчетной сетки отношение производительности GPU к производительности CPU растет.

Графические ускорители нового поколения Tesla K40 не просто производительнее, чем Tesla C2070, но и обладают большей памятью, что критически важно для GPGPU-расчетов в области вычислительной гидродинамики. В первую очередь, именно ограничение со стороны памяти GPU вынуждает организовывать параллельные расчеты с использованием нескольких графических ускорителей. В то же время, затраты на обмен данными с GPU приводят к менее эффективному распараллеливанию в случае большого числа вычислительных узлов. Так, при расчетах с одинарной точностью на сетке, содержащей 5 миллионов ячеек, ускорение параллельных расчетов близко к линейному примерно до десяти GPU-узлов, а затем ухудшается, в отличие от расчетов на CPU, показывающих хорошее ускорение и при большем числе процессоров. При больших сетках параллельные расчеты на GPU-узлах выполняются гораздо эффективнее, а ускорение сопоставимо с ускорением расчетов на CPU.

Расчеты с двойной точностью требуют больше памяти, и соответственно насыщение наступает раньше (около семи GPU для сетки, содержащей 5 миллионов ячеек). В остальном для расчетов с двойной точностью справедливы те же самые выводы, что и для расчетов с одинарной точностью. Самый существенный недостаток при расчетах с двойной точностью на GPU связан с операциями интегрирования,

эффективно реализованными с помощью атомарных операций CUDA в случае одинарной точности представления вещественных чисел.

Работа выполнена с использованием ресурсов суперкомпьютерного комплекса МГУ имени М. В. Ломоносова [22, 23] при финансовой поддержке гранта № 14.Z50.31.0003 Правительства РФ для государственной поддержки научных исследований, проводимых под руководством ведущих ученых в российских ВУЗах (ведущий ученый — С. А. Исаев, Казанский национальный исследовательский технический университет им. А. Н. Туполева).

СПИСОК ЛИТЕРАТУРЫ

1. http://parallel.ru/news/top500_47edition.html
2. https://www.top500.org/lists/2016/06/
3. http://parallel.ru/cluster
4. *Бикулов Д.А., Сенин Д.С.* Реализация метода решеточных уравнений Больцмана без хранимых функций распределения для GPU // Вычислительные методы и программирование: новые вычислительные технологии. 2013. **14**. 370–374.
5. *Маслий А.Н., Мадиров Э.И.* Сравнение производительности квантово-химических расчетов при использовании GPU // Вестник Казанского технологического университета. 2013. **16**, № 23. 12–18.
6. *Юнусов А.А., Губайдуллин И.М., Файзуллин М.Р.* Анализ алгоритмов решения задач химической кинетики с использованием GPGPU // Журнал СВМО. 2010. **12**, № 3. 146–152.
7. *Горобец А.В., Суков С.А., Железняков А.О., Богданов П.Б., Четверушкин Б.Н.* Применение GPU в рамках гибридного двухуровневого распараллеливания MPI+OpenMP на гетерогенных вычислительных системах // Тр. международной конференции “Параллельные вычислительные технологии”. Челябинск: Южно-Уральский гос. ун-т, 2011. 452–460.
8. *Волков К.Н., Дерюгин Ю.Н., Емельянов В.Н., Карпенко А.Г., Козелков А.С., Тетерина И.В.* Методы ускорения газодинамических расчетов на неструктурированных сетках. М.: Физматлит, 2013.
9. *Гаврилов А.А., Минаков А.В., Дектерев А.А., Рудяк В.Я.* Численный алгоритм для моделирования ламинарных течений в кольцевом канале с эксцентриситетом // Сибирский журнал индустриальной математики. 2010. **13**, № 4. 3–14.
10. *Гаврилов А.А., Минаков А.В., Дектерев А.А., Рудяк В.Я.* Численный алгоритм для моделирования установившихся ламинарных течений неньютоновских жидкостей в кольцевом зазоре с эксцентриситетом // Вычислительные технологии. 2012. **17**, № 1. 44–56.
11. *Mavriplis D.J.* Revisiting the least-squares procedure for gradient reconstruction on unstructured meshes // AIAA-Paper 2003-3986. 2003.
12. *Ferziger J.H., Peric M.* Computational methods for fluid dynamics. Heidelberg: Springer, 2002.
13. *Moukalled F., Darwish M.* A unified formulation of the segregated class of algorithms for fluid flow at all speeds // Numerical Heat Transfer. Part B. 2000. **37**, N 2. 227–246.
14. *Белов И.А., Исаев С.А., Коробков В.А.* Задачи и методы расчета отрывных течений несжимаемой жидкости. Л.: Судостроение, 1989.
15. *Патанкар С.* Численные методы решения задач теплообмена и динамики жидкости. М.: Энергоатомиздат, 1984.
16. *Rhie C.M., Chow W.L.* Numerical study of the turbulent flow past an airfoil with trailing edge separation // AIAA Journal. 1983. **21**, N 11. 1525–1532.
17. *Leonard B.P.* A stable and accurate convective modelling procedure based on quadratic upstream interpolation // Comp. Math. Appl. Mech. Eng. 1979. **19**, N 1. 59–98.
18. *Barrett R., Berry M.W., Chan T.F., et al.* Templates for the solution of linear systems: building blocks for iterative methods. Philadelphia: SIAM, 1994.
19. *Самарский А.А., Николаев Е.С.* Методы решения сеточных уравнений. М.: Наука, 1978.
20. *Karypis G., Kumar V.* A fast and high quality multilevel scheme for partitioning irregular graphs // SIAM Journal on Scientific Computing. 1998. **20**, N 1. 359–392.
21. *Гаврилов А.А., Кривов М.А., Гризан С.А., Дектерев А.А.* GPU версия CFD пакета SigmaFlow: портирование и оптимизация с использованием инструментария TTG APPTIMIZER // Тр. международной научной конференции “Параллельные вычислительные технологии”. Челябинск: Южно-Уральский гос. ун-т, 2013. 106–115.
22. *Воеводин Вл.В., Жуматий С.А., Соболев С.И., Антонов А.С., Брызгалов П.А., Никитенко Д.А., Стефанов К.С., Воеводин Вад.В.* Практика суперкомпьютера “Ломоносов” // Открытые системы. 2012. № 7. 36–39.
23. *Sadovnichy V., Tikhonravov A., Voevodin Vl., Opanasenko V.* “Lomonosov”: Supercomputing at Moscow State University // Contemporary High Performance Computing: From Petascale toward Exascale. Boca Raton: CRC Press, 2013. 283–307.

Поступила в редакцию
22.07.2016

Efficiency Analysis of Hydrodynamic Calculations on GPU and CPU Clusters

A. V. Sentyabov [1], A. A. Gavrilov [2], M. A. Krivov [3], A. A. Dekterev [4], and M. N. Pritula [5]

[1] *Kutateladze Institute of Thermophysics, Siberian Branch of the Russian Academy of Sciences; prospekt Lavrent'eva 1, Novosibirsk, 630090, Russia; Ph.D., Junior Scientist, e-mail: sentyabov_a_v@mail.ru*

[2] *Kutateladze Institute of Thermophysics, Siberian Branch of the Russian Academy of Sciences; prospekt Lavrent'eva 1, Novosibirsk, 630090, Russia; Ph.D., Leading Engineer, e-mail: gavand@yandex.ru*

[3] *TTG Labs Limited Liability Company, Skolkovo Innovation Center; ulitsa Nobelya 7, Moscow, 143026, Russia; Chief Executive Officer, e-mail: m_krivov@ttgLabs.com*

[4] *Kutateladze Institute of Thermophysics, Siberian Branch of the Russian Academy of Sciences; prospekt Lavrent'eva 1, Novosibirsk, 630090, Russia; Ph.D., Senior Scientist, e-mail: dekterev@mail.ru*

[5] *TTG Labs Limited Liability Company, Skolkovo Innovation Center; ulitsa Nobelya 7, Moscow, 143026, Russia; Ph.D., Developer, e-mail: m_pritula@ttgLabs.com*

Received July 22, 2016

Abstract: Speedup of parallel hydrodynamic calculations on clusters with CPUs and GPUs is considered. The CFD SigmaFlow code developed by the authors and ported for GPU by means of CUDA is used in test calculations. The incompressible flow simulation is based on a SIMPLE-like procedure and on a discretization by the control volume method on unstructured hexahedral meshes. The performance evaluation shows a high efficiency of the new generation of GPUs for GPGPU calculations.

Keywords: GPGPU, numerical simulation, computational fluid dynamics, SIMPLE, MPI, CUDA.

References

1. Supercomputing News, Parallel.ru. http://parallel.ru/news/top500_47edition.html. Cited August 30, 2016.

2. Top 500. https://www.top500.org/lists/2016/06/. Cited August 30, 2016.

3. Supercomputer Center of Moscow University. http://parallel.ru/cluster. Cited August 30, 2016.

4. D. A. Bikulov and D. S. Senin, "Implementation of the Lattice Boltzmann Method without Stored Distribution Functions on GPU," Vychisl. Metody Programm. **14**, 370–374 (2013).

5. A. N. Maslii and E. I. Madirov, "Performance Comparison of Quantum-Chemical Calculations Using GPU," Vestn. Kazan Tekhnol. Univ. **16** (23), 12–18 (2013).

6. A. A. Yunusov, I. M. Gubaydullin, and M. R. Fayzullin, "Analysis of Algorithms for Solving Chemical Kinetics Problems Using GPGPU," Zh. Srednevolzh. Matem. Obshchestva **12** (3), 146–152 (2010).

7. A. V. Gorobets, S. A. Sukov, A. O. Zheleznyakov, et al., "Application of GPU for Hybrid Two-Level Parallelization MPI+OpenMP on Heterogeneous Computing Systems," in *Proc. Int. Conf. on Parallel Computational Technologies, Moscow, Russia, March 28–April 1, 2011* (South Ural State Univ., Chelyabinsk, 2011), pp. 452–460.

8. K. N. Volkov, Yu. N. Deryugin, V. N. Emel'yanov, A. G. Karpenko, A. S. Kozelkov, and I. V. Teterina, *Methods for Accelerating Gasdynamic Calculations on Unstructured Grids* (Fizmatlit, Moscow, 2013) [in Russian].

9. A. A. Gavrilov, A. V. Minakov, A. A. Dekterev, and V. Ya. Rudyak, "A Numerical Algorithm for Modeling Laminar Flows in an Annular Channel with Eccentricity," Sib. Zh. Ind. Mat. **13** (4), 3–14 (2010) [J. Appl. Ind. Math. **5** (4), 559–568 (2011)].

10. A. A. Gavrilov, A. V. Minakov, A. A. Dekterev, and V. Ya. Rudyak, "Numerical Algorithm for Fully Developed Laminar Flow of a Non-Newtonian Fluid through an Eccentric Annulus," Vychisl. Tekhnol. **17** (1), 44–56 (2012).

11. D. J. Mavriplis, "Revisiting the Least-Squares Procedure for Gradient Reconstruction on Unstructured Meshes," AIAA Paper 2003-3986 (2003).

12. J. H. Ferziger and M. Peric, *Computational Methods for Fluid Dynamics* (Springer, Heidelberg, 2002).

13. F. Moukalled and M. Darwish, "A Unified Formulation of the Segregated Class of Algorithms for Fluid Flow at All Speeds," Numer. Heat Transfer. Part B. **37** (2), 227–246 (2000).

14. I. A. Belov, S. A. Isaev, and V. A. Korobkov, *Problems and Methods of Calculation of Separating Flows of Incompressible Fluids* (Sudostroenie, Leningrad, 1989) [in Russian].

15. S. Patankar, *Numerical Heat Transfer and Fluid Flow* (Hemisphere, New York, 1980; Energoatomizdat, 1984).

16. C. M. Rhie and W. L. Chow, "Numerical Study of the Turbulent Flow Past an Airfoil with Trailing Edge Separation," AIAA J. **21** (11), 1525–1532 (1983).

17. B. P. Leonard, "A Stable and Accurate Convective Modelling Procedure Based on Quadratic Upstream Interpolation," Comput. Methods Appl. Mech. Eng. **19** (1), 59–98 (1979).

18. R. Barrett, M. W. Berry, T. F. Chan, et al., *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods* (SIAM, Philadelphia, 1994).

19. A. A. Samarskii and E. S. Nikolaev, *Numerical Methods for Grid Equations* (Nauka, Moscow, 1978; Birkhäuser, Basel, 1989).

20. G. Karypis and V. Kumar, "A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs," SIAM J. Sci. Comput. **20** (1), 359–392 (1998).

21. A. A. Gavrilov, M. A. Krivov, S. A. Grizan, and A. A. Dekterev, "GPU Version of CFD Software SigmaFlow: Porting and Optimization Using TTG Apptimizer Toolkit," in *Proc. Int. Conf. on Parallel Computational Technologies, Chelyabinsk, Russia, April 1–5, 2013* (South Ural State Univ., Chelyabinsk, 2013), pp. 106–115.

22. Vl. V. Voevodin, S. A. Zhumatii, S. I. Sobolev, et al., "The Lomonosov Supercomputer in Practice," Otkrytye Sistemy, No. 7, 36–39 (2012).

23. V. Sadovnichy, A. Tikhonravov, Vl. Voevodin, and V. Opanasenko, "'Lomonosov': Supercomputing at Moscow State University," in *Contemporary High Performance Computing: From Petascale toward Exascale* (CRC Press, Boca Raton, 2013), pp. 283–307.