1 Тема

Разработка подхода к созданию мультиплатформенного расчетного кода для численного моделирования сжимаемых течений, 2017-2018 гг.

2 Состав коллектива

Шершнёв Антон Алексеевич, к.ф.-м.н., н.с., ИТПМ СО РАН
Кашковский Александр Владимирович, к.т.н., с.н.с., ИТПМ СО РАН

3 Информация о грантах

РФФИ 16-57-48007, «Разработка высокопроизводительных компьютерных кодов и их применение для численного моделирования в аэродинамике и динамике разреженных газов», руководитель А.Н. Кудрявцев

4 Научное содержание работы

4.1 Аннотация

Многие задачи вычислительной аэродинамики сверхзвуковых течений требуют использования трехмерных сеток с большим разрешением. Одним из примеров таких задач является моделирование ламинарно-турбулентного перехода. Одним из возможных способов уменьшения времени вычислений является использование гибридных кластеров, содержащих как центральные процесорные устройства (ЦПУ), так и графические процессорные устройства (ГПУ). Обычно, ГПУ используются в либо качестве сопроцессоров и выполняют лишь отдельные этапы численного алгоритма, либо в качестве основных вычислительных устройств и выполняют все этапы алгоритма, а ЦПУ при этом используются только для управления графическими устройствами. В тоже время наиболее эффективным было бы использование гибридного кластера в гетерогенном режиме, то есть когда и ЦПУ, и ГПУ являются равноправными вычислительными устройствами.

В данной работе рассмотрен подход к разработке вычислительного кода для численного моделирования сжимаемых высокоскоростных течений на основе нестационарных уравнений Навье-Стокса, который позволяет использовать его различных типах вычислительных устройств, в том числе ГПУ и многоядерных ЦПУ. Основная идея заключается в использовании специального интерфейса, позволяющей получить из одного и того же исходного кода несколько исполняемых файлов под каждую из вычислительных платформ. Созданный код верифицирован, проведено сравнение его вычислительной эффективности в расчетах на ЦПУ и ГПУ.

4.2 Постановка задачи

Основной идей создания гибридного кода является адаптация программ написанных для выполнения на ГПУ с использованием технологии CUDA для вычислений на ЦПУ или Xeon Phi с использованием технологии OpenMP. Многоядерный ЦПУ можно представить в виде одного блока CUDA, в котором число нитей равно числу ЦПУ ядер. Эти нити создаются с помощью OpenMP. Для каждого OpenMP потока вызывается аналог CUDA kernel. Исходный текст должен быть одинаков для всех платформ. Для модификации исходного текста под конкретную платформу используется препроцессор языка C/C++. Все платформозависимые выражения заменяются директивами препроцессора. Желательно уменьшить число директив препроцессора. Часть функций может быть переписана индивидуально под каждую платформу, но таких функций должно быть как можно меньше. В случае гетерогенных вычислений нет необходимости в едином коде для всех платформ. Исполняемые файлы для разных платформ легко объединяются с помощью MPI. Но необходим баланс загрузки, который равномерно распределит вычислительную нагрузку между всеми вычислительными устройствами.

4.3 Результаты

Апробация предложенной методики проводилась на примере численного кода HyCFS, изначально разработанного для расчетов с использованием ГПУ (см. отчет за предыдущий отчетный период). Программа решает полные нестационарные уравнения Навье-Стокса. Для вычисления конвективных членов используется WENO схема Шу и Ошера сквозного счета 5-го порядка точности. Их вычисление занимает примерно 80% всего времени вычисления. Диффузионные члены вычисляются с помощью центральных разностей 2-го порядка точности. Интегрирование по времени выполняется явной схемой Рунге-Кутты-Гилла 4-го порядка точности. Граничные условия ставятся с помощью нескольких рядов фиктивных ячеек. Программа ориентирована на кластеры, у которых на каждом узле имеется несколько ГПУ. Параллельное управление ГПУ осуществляется с помощью OpenMP, а передача данных между узлами с помощью MPI (см. рис. 1. Так как при вычислениях на ЦПУ желательное использовать все OpenMP потоки, а ГПУ не используются, управляющие ими OpenMP потоки были отключены за ненадобностью.

Всего исходный текст содержал порядка 13 тыс. строк. При адаптации были изменны 327 строк, что составляет $\sim$2.5%. Таким образом, изменения в программы были минимальными.

Рассматривалось два тестовых случая:

1. Размер сетки $200 \times 100 \times 420 = 8.4$ million of cells. Расчеты проводились с использованием WENO scheme 5-го порядка.

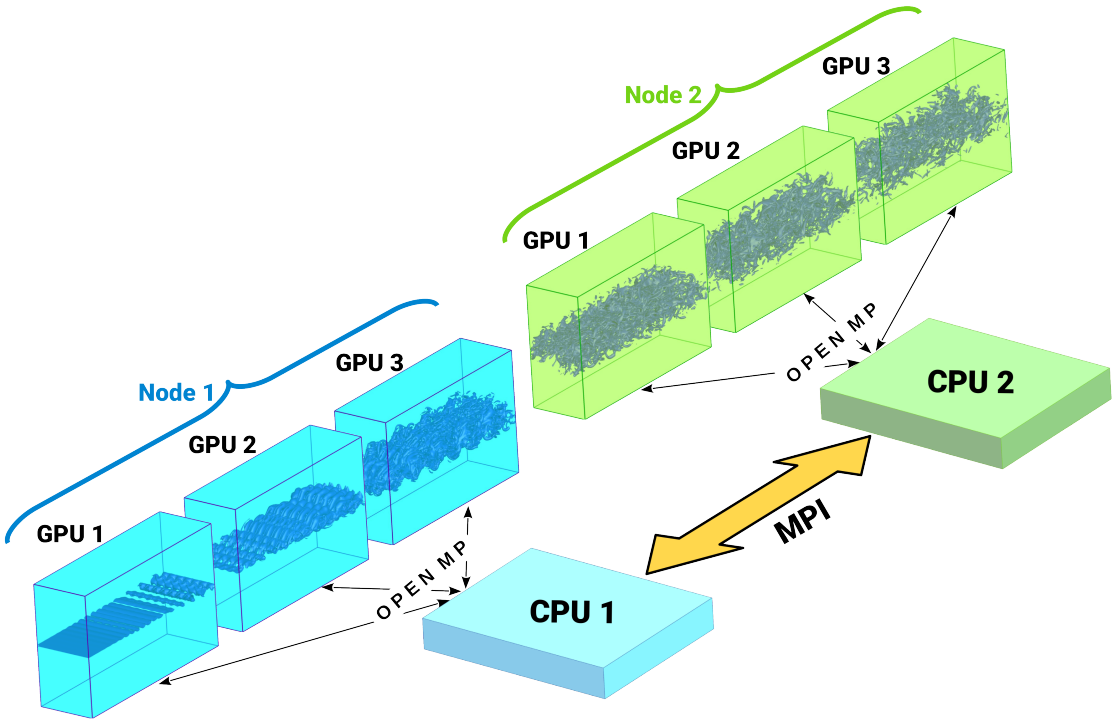

Рис. 1: Схема параллелизации HyCFS.

2. Размер сетки $300 \times 100 \times 480 = 14.4$ million of cells with TVD scheme 1-го порядка.

Расчеты проводились в информационно-вычислительном центре Новосибирского государственного университета. Использовались следующие вычислительные ресурсы:

- GPU, NVIDIA Tesla M2090, 6 ГБ памяти GDDR5 с пропускной способностью 177 ГБ/сек, 512 ядер, 665 Гфлопс пиковой производительности для вычислений двойной точности.
- CPU, Два 6-ядерных процессора Intel Xeon X5670 с тактовой частотой 2933 МГц,
- CPU, Два 12-ядерных процессора Intel Xeon E5-2680v3 с тактовой частотой 2500 МГц
- Intel Xeon Phi 7120P (второе поколение процессоров на базе архитектуры Intel MIC, кодовое название - 'Knights Corner'), 16 ГБ собственного ОЗУ, 61 ядро, работающее в 244 потока на частотах 1.2 - 1.3 ГГц 1.2 Тфлопс пиковой производительности.

В таблице 1 представленно среднее время выполнения одного шага для указнных тестовых случаев.

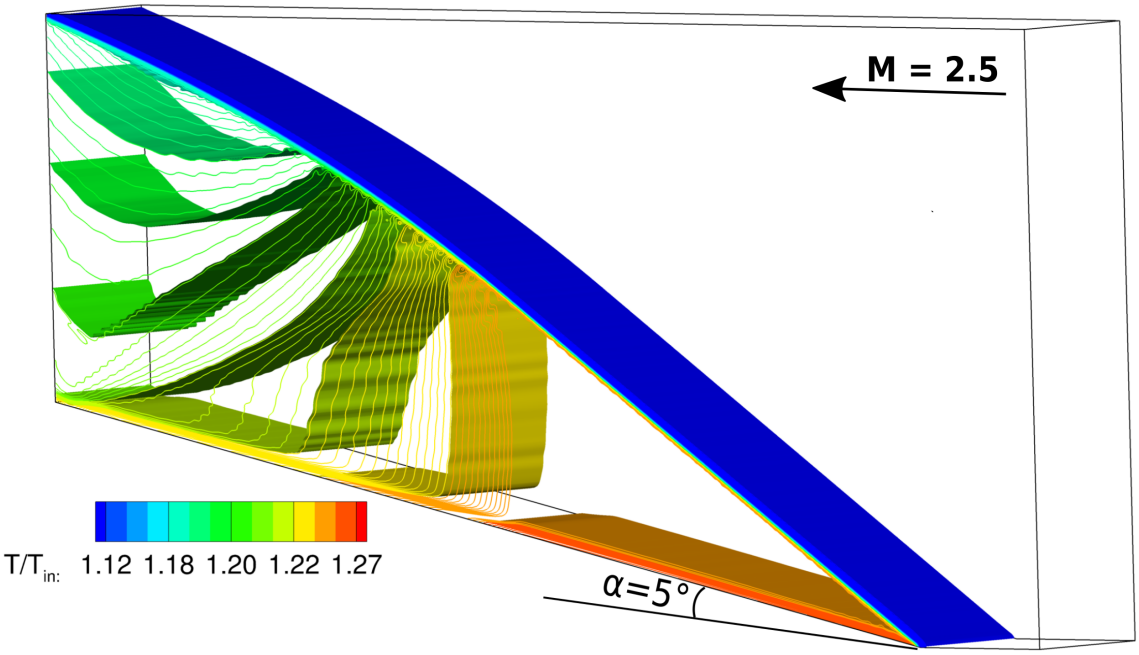

Рис. 2: Тестовый случай

Таблица 1: Сравнение производительности

Device	Time [s/step]	
	Case 1	Case 2
NVIDIA Tesla M2090	6.10	2.42
Intel Xeon X5670 / 2 × 6 cores	16.92	-
Intel Xeon E5-2680v3 / 1 × 12 cores	15.70	-
Intel Xeon E5-2680v3 / 2 × 12 cores	10.40	5.66
Intel Xeon Phi 7120P	34.35	11.65

Видно, что для данных тестовых случаев, GPU показало наименьше время вычислений. Использование 12 ядер на разных ЦПУ показало примерно одинаковое время вычислений, уступая ГПУ примерно в 2.5 раза. Однако, двухкратное увеличение числа ядер на ЦПУ E5-2680v3 привело лишь к ускорению в 1.5 раза (а не в 2, как ожидалось). Возможно, это произошло из-за снижения вычислительной нагрузки на каждое ядро, что привело к неоптимальному разделению загрузки. Тем не менее, можно предположить, что время вычисления на одном ГПУ Tesla M2090 примерно эквивалентно 30-40 ядрам ЦПУ E5-2680v3.

Совершенно неожиданный результат получился для Xeon Phi 7120P. Несмотря на существенно большую заявленную пиковую производительность, вычисления на нем почти в 5 раз медленне чем на ГПУ. Даже по сравнению с ЦПУ, используя в 5 раз большее число ядер, вычисления оказались в 2 раза медленее! Такую разницу трудно объяснить неоптимальностью программы, или настрой-

ками транслятора и процессора. Более детальное исследование производительности свидетельствует, что, по всей видимости, это особенности архитектуры устройства, которое, похоже, показывает пиковую производительность лишь в очень специфических задачах и синтетических тестах. Расчеты на следующем поколении устройств ('Knights Landing') показали достаточно высокую производительность, сравнимую с производительностью ГПУ или 20-30 ядрами ЦПУ.

5 Эффект от использования кластера в достижении целей работы

Разработка многоплатформенного кода естественным образом требует наличия вычислительных устройств с различной архитектурой. Поскольку в составе кластера НГУ имеются все наиболее распространенные в настоящее время вычислительные архитектуры (многоядерные ЦПУ, ГПУ и сопроцессоры Intel Xeon Phi), его использования позволило разработать требуемую методику, провести тестирование и провести измерения эффективности параллелизации в различных конфигурациях.

6 Список публикаций

1. Alexander V. Kashkovsky, Anton A. Shershnev, and Pavel V. Vashchenkov, Aspects of GPU perfomance in algorithms with random memory access, AIP Conference Proceedings 1893, 030047 (2017);

   `https://doi.org/10.1063/1.5007505`

2. A. V. Kashkovsky, A. A. Shershnev, A. N. Kudryavtsev, and D. V. Khotyanovsky, Approach to the development of a multiplatform code for numerical simulation of compressible flows, AIP Conference Proceedings 1893, 030048 (2017);

   `https://doi.org/10.1063/1.5007506`

3. Alexey N. Kudryavtsev, Alexander V. Kashkovsky, Semyon P. Borisov, and Anton A. Shershnev, A numerical code for the simulation of non-equilibrium chemically reacting flows on hybrid CPU-GPU clusters, AIP Conference Proceedings 1893, 030054 (2017);

   `https://doi.org/10.1063/1.5007512`

4. Anton A. Shershnev, Georgy V. Shoev, Semyon P. Borisov, Alexander V. Kashkovsky, and Alexey N. Kudryavtsev, Numerical approach for the simulation of thermally non-equilibrium dissociating flows on hybrid supercomputers, AIP Conference Proceedings 2027, 030042 (2018);

   `https://doi.org/10.1063/1.5065136`

5. S. P. Borisov, A. N. Kudryavtsev, and A. A. Shershnev, Numerical simulation of detonation propagation in a plane channel with detailed chemical mechanisms on supercomputers with GPUs, AIP Conference Proceedings 2027, 040023 (2018);

   https://doi.org/10.1063/1.5065297

6. Yu. V. Kratova, A. V. Kashkovsky, and A. A. Shershnev Numerical simulation of heterogeneous detonation in polydisperse gas suspensions using modern parallel computational architectures, AIP Conference Proceedings 2027, 040086 (2018); doi: 10.1063/1.5065360

   https://doi.org/10.1063/1.5065360