Оптимизация для кластера с ускорителями Xeon Phi задачи фильтрационного течения жидкостей в трехмерных средах с двойной пористостью

А.С. Первунин

Аннотация - на основе ранее разработанного программного комплекса для моделирования многофазных потоков в деформируемой пористой среде с использованием итеративных методов решения дифференциальных уравнений Рунге-Кутта 5-го порядка был реализован параллельный программный комплекс, оптимизированный для запуска на существующем кластере с ускорителями Intel Xeon Phi, являющимися аналогами графических ускорителей в виде многоядерных сопроцессоров. Хотя данные ускорители уже перестали производиться, методы, изложенные в данной статье, могут обеспечить повышение быстродействия и на преемниках этого процессора.

Были рассмотрены различные способы оптимизации программного кода, описанные в данной статье, которые специфичны для данного ускорителя, и их влияние на время работы вычислительной программы. Произведено сравнение различных способов использования ускорителей в составе кластера: родного режима, симметричного режима и разгрузочного режима. Также было приведено краткое описание данных режимов. Было показано, что при использовании многоядерного сопроцессора Intel Xeon Phi возможно получить значительное ускорение при выполнении расчётов. Получены оценки ускорения и эффективности при использовании различного числа узлов вычислительного кластера при использовании программного комплекса. Перечислены параметры целевого вычислительного кластера.

Ключевые слова: Метод Рунге-Кутта, высокопроизводительные вычисления, параллельное программирование, оптимизация программ, ускорители Intel Xeon Phi.

I. Введение

В работе [1] была представлена модель фильтрации двухфазной жидкости через деформируемый пористый каркас, и предложен метод, основанный на алгоритме Рунге-Кутты. На основе этих результатов автором была создана двухмерная реализация данной модели для двух жидкостей для моделирования фильтрации водно-нефтяной смеси через пористую среду. В рамках дальнейшей работы задача была реализована с использованием метода Рунге-Кутты 5-го порядка точности по времени.

Данная работа целиком посвящена вопросу параллельной реализации и оптимизации данной задачи для кластера, использующего ускорители Intel Xeon Phi. Статья имеет структуру, состоящую из двух разделов. Первый раздел содержит описание задачи: приводится система решаемых уравнений, используемые методы ее решения и схема численного алгоритма.

Перечислены параметры целевой вычислительной системы. Второй раздел содержит описание используемых методов оптимизации и распараллеливания программы. Приведены оценки степени ускорения, достигнутого на каждом этапе оптимизации и степени распараллеливания.

По своей архитектуре многоядерные сопроцессоры Intel Xeon Phi являются концептуальными аналогами (заменой) графических ускорителей. Хотя данные ускорители уже перестали выпускаться, методы, изложенные в данной статье, могут обеспечить повышение быстродействия и на преемниках этого процессора.

При использовании многоядерного сопроцессора Intel Xeon Phi возможно получить значительное ускорение при выполнении расчётов. Intel Xeon Phi имеет поддержку 3 режима работы — «родной», «разгрузочный» и «симметричный».

Было проведено сравнение двух режимов работы сопроцессоров Xeon Phi: симметричного и режима «разгрузочный». В разгрузочном режиме параллельная часть программы производит копирование из памяти вычислительного узла в память сопроцессора и вычисления происходят на сопроцессоре. В «симметричном»-режиме вычислительную нагрузку берут на себя как CPU, так и сопроцессор. Многоядерные сопроцессоры поддерживают технологии параллельного программирования OpenMP и MPI.

В заключении приводятся основные полученные результаты.

II. Описание задачи

A. *Вычислительная модель задачи*

Система уравнений движения смеси жидкостей через эластичную среду, представленная в [1], выглядит следующим образом: —

$$\frac{\partial \rho\alpha_n}{\partial t} + \partial_k(\rho\alpha_n u_k) = -\sum_{m=2}^{N} \lambda_{mn}(p_1 - p_m), n = 2,\ldots,N$$

$$\frac{\partial \alpha_n \rho_n}{\partial t} + \partial_k(\alpha_n \rho_n u_k^n) = 0, n = 2,\ldots,N$$

$$\frac{\partial \omega_k^n}{\partial t} + \partial_k \left(\frac{1}{2}u_i^1 u_i^1 - \frac{1}{2}u_i^n u_i^n + e^1 + \frac{p}{\rho} - e^n - \frac{p_n}{\rho_n}\right) = e_{kij} u_i \omega_j^n - \sum_{m=2}^{N} \chi_{nm} c_m (u_k - u_k^m)$$

$$\frac{\partial \rho F_{ij}}{\partial t} + \partial_k (\rho F_{ij} u_k - \rho F_{kj} u_i) = 0,$$

$$\frac{\partial \rho s}{\partial t} + \partial_k (\rho s u_k) = \frac{1}{T} R,$$

Где

$$R = \sum_{n,m=2}^{N} \chi_{nm} (u_i - u_i^n)(u_i - u_i^m) + \frac{1}{\rho^2} \sum_{n,m=2}^{N} \lambda_{nm} (p_1 - p_n)(p_1 - p_m)$$

— диссипативная функция, коэффициенты α_1 – объемная концентрация эластичного каркаса (фаза 1), α_n – объемные концентрации жидкостей (фаза n, n = 2, ... ,N, где N-1 – число жидкостей в смеси, просачивающейся через пористую среду), причем $\alpha_1 + \sum_{n=2}^{N} \alpha_n = 1$ c_1 – массовая концентрация эластичного каркаса (фаза 1), c_n – массовые концентрации жидкостей (фаза n), причем $c_1 + \sum_{n=2}^{N} c_n = 1$. ρ_n – массовая плотность фазы n, Nn=1 – массовая плотность N-фазной среды. $uuii\ nn$ – скорость фазы n, n= 1,…,N; $w_i\ n = u_i\ 1 - u_i\ n$ – относительная скорость жидкой фазы n по отношению к пористой среде. $u_1 + \sum_{n=2}^{N} u_n = 1$ n=1 – скорость N=фазной среды, где $cc_n = \alpha_n \rho_n / \rho$. Fij – компоненты тензора градиента деформации всей N-фазной среды. s – массовая плотность энтропии всей N-фазной среды. $p + \sum_{n=2}^{N} \alpha_n p_n = 1$ $p_n = p_n^2 e_{p_n}^n$ – фазовые плотности. Эта система дополнена условием согласованности в форме законов сохранения $e_{ikj}\omega_j^n = \partial_i \omega_k^n - \partial_k \omega_i^n$

$$\frac{\partial \omega_k^n}{\partial t} + \partial j(u_j \omega_k^n - u_k \omega_j^n + \sum_{m=2}^{N} e_{kij} \chi_{nm} E \omega_i^m) = 0, \partial_k (pF_{kj}) = 0$$

Для решения задачи в двумерном случае система уравнений была векторизована [1]:

$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} + \frac{\partial G(U)}{\partial y} = S(U)$$

F(U), G(U) – векторы потоков по направлениям x и y, S(U) – вектор правых частей.

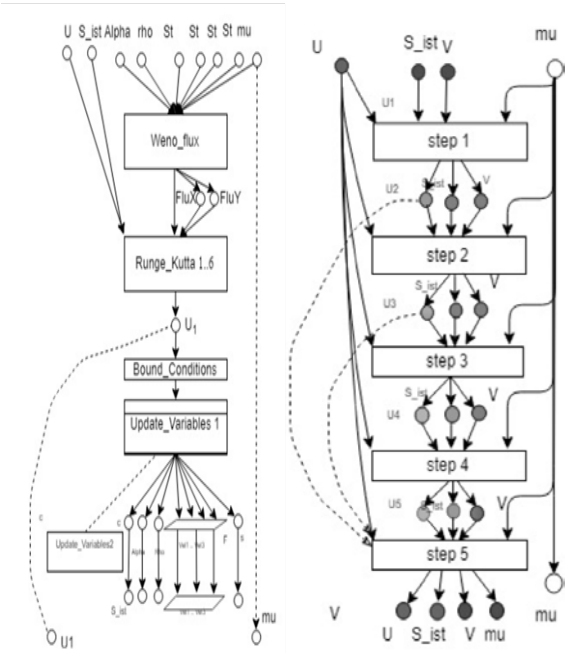
Данная работа рассматривает решение системы уравнений на прямоугольной сетке методом WENO-Рунге-Кутта [1], в котором вычисление потоков через грани ячеек производится с использованием аппроксимации данных полиномами внутри ячеек, что обеспечивает 5-й порядок точности по пространству, а для интегрирования уравнений по времени используется метод Рунге-Кутта с 5-м временным порядком точности. Значение каждого очередного временного шага вычисляется на основе текущей максимальной скорости звука, если исходить из условия Куранта.

B. Схема численного алгоритма

Алгоритмическая схема представлена в виде графа зависимостей с переменными единственного присваивания (черные кружки) и операциями единственного срабатывания (серые прямоугольники). Каждая переменная является одной или несколькими распределенными в пространстве характеристиками среды, реализуемыми в программе как массивы размерностью два. Операции отражают порядок вычисления одних массивов из других. Сплошные стрелки обозначают информационные зависимости, штриховые стрелки обозначают тождественность соединяемых переменных.

Особенностью данного алгоритма является то, что все проведенные вычисления внутри ячеек осуществляемые в пределах каждой отдельной операции данного алгоритма (вложенных циклов) не зависят друг от друга, хотя СС и используют значения в соседних ячейках. Это позволяет легко преобразовать данный алгоритм в параллельный при помощи метода декомпозиции в пространстве моделирования. Проведенный анализ графа, который представлен на рис. 1а позволил выявить самый оптимальный этап данного алгоритма, на котором эффективнее всего будет выполняться обмен граничными значениями при случае применения пространственной декомпозиции.

На рис. 1б представлена схема одного шага по времени для данного численного алгоритма. (б)



(а) *(б)*

Рис. 1. (а) Общая схема вычислительного алгоритма, реализующего метод Рунге-Кутта 5-го порядка точности (б) Схема одного шага метода Рунге-Кутта

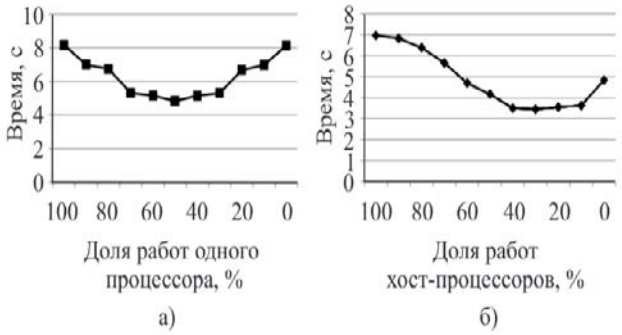


Рис. 2. Время вычисления одного временного шага для сетки с размерность 1000 при использовании двух сопроцессоров с различными соотношениями работ между ними (а), при задействовании всех ресурсов узла с различным соотношением работ между хост-процессорами и сопроцессорами, а также равномерным распределением работ между сопроцессорами (б)

В алгоритме используются шесть операций Step_1…Step_6, каждая из которых соответствуют своей стадии вычислений в методе Рунге-Кутта с 5-м порядком точности. Логика вычислений, происходящая на всех шести стадиях совпадает почти полностью и может быть вынесена в отдельную подпрограмму, схема которой показана на рис. 1б. Переменные, которые на данной схеме также представляют собой распределенные в пространстве параметры среды, которые реализованы в программе как массивы размерностью 2, а каждая из операций была реализована как набор циклов с вложенностью, выполняющие обход по ячейкам данной пространственной сетки и вычисляющая значения выходных характеристик для данной ячейки. Процедура WENO_Flux производит вычисления значения потоков по границам ячеек по значениям примитивных переменных. Операция Runge-Kutta_1..6 вычисляет новые значения накопительных переменных на основе значений потоков по формулам, которые специфичны для данного этапа метода Рунге-Кутта. Операция Bound_Conditions производит обновление значения накопительных переменных на границах рассматриваемой области. Операции Update_Variables_1 и Update_Variables_2 производят вычисления значений примитивных переменных из накопительных.

Описанный алгоритм был реализован в виде последовательного программного комплекса на языке Си и послужил основой для настоящей работы.

C. Возможности целевой вычислительной системы

Основной целью данной работы являлось оптимизация комплекса программ для работы на гибридном вычислительном комплексе ССКЦ СО РАН, каждый узел которого, кроме основных процессоров (хост-процессоров) содержит два ускорителя с архитектурой Intel Many Integrated Cores. В качестве хост-процессоров используются два 8-ядерных процессора Intel Xeon E5-2697 2.9 ГГц с поддержкой технологии Hyperthreading, что в сумме дает 32 аппаратных потока и суммарную пиковую производительность 85 TFLOPS на каждый узел. Оперативная память, установленная на каждом узле, составляет 64 Гб. В качестве сопроцессоров используются два ускорителя Intel Xeon Phi 7110X 1.24 ГГц, каждый из которых имеет 61 ядро, поддерживающее 4 аппаратных потока, что в сумме дает 244 аппаратных потока и около 1.3 TFLOPS пиковой производительности на ускоритель для одинарной точности и 0.6 TFLOPS для двойной. Объем оперативной памяти, доступной каждому ускорителю, составляет 6 Гб. Производительность элементарных операций, выполняемых на данном кластере оценивалась с помощью набора тестов. Использовавшееся для работы программное обеспечение: Intel C/C++ Compiler, Intel, MPI Library.

III. Оптимизация алгоритма для кластера с ускорителями Xeon Phi

A. Оптимизация последовательной программы

Основными моментами для достижения высокой производительности на ускорителях Xeon Phi являются наличие у них на борту большого количества ядер и поддержка ими векторных операций с длинными (64-байтными) векторами.

С целью векторизации вычислений были проведен набор следующих действий:

• *Во всех вложенных циклах, обозначающих обход пространственной сетки перед внутренним циклом (по строке) была добавлена директива компилятора #pragma ivdep, сигнализирующая компилятору об отсутствии взаимных зависимостей между итерациями циклов.*

• *В операции weno_flux (weighted essentially non-oscillatory method), которая выполняет вычисление потоков через границы ячеек, тело цикла оказалось слишком большим, поэтому компилятор отказался от его оптимизации. Данная ситуация была разрешена путем разбиения одной операции weno_flux на шесть операций (гнезд циклов) меньшего размера, что потребовало использования 32-х дополнительных двумерных массивов.*

• *В операции update_variables_1, которая выполняет начальный этап вычисления примитивных переменных из накопительных, чтобы стабилизировать давления в ячейке происходит коррекция объемных концентраций жидких фаз с помощью метода Ньютона. Число итераций метода Ньютона, который реализуется с помощью цикла типа while, имеет зависимость от состояния конкретной ячейки. В данном случае цикл по пространству, который содержит в своем теле другой цикл, не был векторизован. Чтобы оптимизировать всё гнездо циклов по пространству было разбито на субгнезда меньшего размера, содержащие:*

1) операции двух первых итераций цикла while (выполняются в любом случае)

2) операции, выполняющиеся за пределами цикла while

3) все остальные итерации цикла while

Таким образом, была проведена векторизация первого и третьего «гнезда» циклов.

Каждое из вышеперечисленных действий сократило время вычисления. В целом векторизация (распараллеливание, при котором команды модифицируются для выполнения нескольких однотипных операций одновременно) всего текста программы придало ускорение на Xeon Phi в 2 раза, на хост-процессорах – на 10%.

Столь малое ускорение объяснятся, тем что основное время алгоритма тратится на выполнение итераций в методе Ньютона.

Выравнивание доступа в памяти означает, что все последовательные обращения к массивам данных внутри векторизованного цикла происходят по адресам, которые кратны единичному размеру вектора. Обеспечение данного условия для компилятора позволяет применять в цикле более быстрые инструкции выровненного данным конкретным образом обращения в оперативную память.

Данное условие было достигнуто путем:

• выравнивания выделения памяти с помощью функции _mm_malloc (или posix_memalign);

• выравнивания размера строчек всех массивов по размеру, который кратен размеру вектора;

• спецификации компилятору внутри векторизованных циклов с помощью встроенной функции имеющейся у компилятора _assume_aligned, что данный массив правильным образом выровнен в памяти.

В результате выравниваний обращений в память программа получила ускорение на Xeon Phi всего на 1,5%, на хост-процессорах – на 0.4%. Такая малая степень ускорения объясняется тем фактом, что самая значительная доля времени в программе расходуется не на обращения в память, а на однотипные вычисления, которые данная оптимизация не затрагивает.

Также еще одним приёмом оптимизации, рекомендуемым при использовании с ускорителями Xeon Phi, является использование метода потоковой записи в ячейки памяти. При использовании такой оптимизации запись элементов, вычисляемых в циклах массивов, происходит сразу же напрямую в оперативную память, минуя при этом кэш-память. Данный результат был достигнут с помощью добавления до векторизованных циклов директивы #pragma vector nontemporal. В результате этого на Xeon Phi программа ускорилась очень незначительным образом, а на хост-процессорах наблюдалось замедление на 30%.

Упоминая способы оптимизации данной программы, следует упомянуть о стандартных методах и приемах оптимизации, таких как вынесение общих подвыражений и замена долго выполняющихся операций на быстро выполняющиеся. В большинстве случаев компилятор не имеет права на выполнение преобразований вещественных выражений с целью оптимизации имеющими дело с вещественными числами. Подобный вид оптимизаций остается на полной ответственности программиста [5]. С помощью преобразования вычисляемых выражений, проведенных вручную, рассматриваемая программа получила ускорение более чем в 2 раза как на Xeon Phi, так и на хост-процессоре, а также с помощью директив препроцессора OpenMP.

В таблице 1 было приведено сравнение времен работы последовательной программы до и после проведения всего цикла оптимизаций. Видно, что на данной программе, оптимизированной последовательным образом ускоритель Xeon Phi значительно проигрывает центральному процессору. Этот результат вполне ожидаем, с учетом того, что ядра Xeon Phi значительно более просты архитектурно, а одним из главных способов оптимизации для них – векторизацией – на данной задаче принес незначительный выигрыш по производительности.

Таблица 1. Время выполнения одного шага по времени для сетки размером 100×100

	Хост-процессор	*Intel Xeon Phi*
Неоптимизированная программа	1.68	28.15 с
Оптимизированная программа	0.78	8.82 с
Ускорение	2.3	3,2

B. Распараллеливание внутри одного вычислительного узла

Каждый отдельный вычислительный узел кластера не учитывая сопроцессоры, так же, как и каждый со-процессор Xeon Phi в отдельности, можно рассматривать как многоядерную систему, имеющую общую память. Распараллеливание задач по ядрам является вторым из главных способов оптимизации программ для ускорителей Intel Xeon Phi. Использовать все ядра такой вычислительной системы для решения одной задачи возможно с помощью технологии директив препроцессора OpenMP. В рассматриваемой случае таким способом были распараллелены внешние циклы во всех наборах вложенных циклов, которые обозначают обход пространственной сетки. В данной задаче число параллельных потоков имеет прямую зависимость от размера задачи, поэтому на задачах малой размерности часть ядер графического процессора у ускорителя Xeon Phi будет простаивать.

На основе результатов, приведенных на рис. 2, возможно сделать вывод, что на хост-процессорах узла было получено девятикратное максимальное ускорение на 32 потоках. На отдельно взятом Xeon Phi максимальное ускорение составило 104,5 раза при использовании 183 потоков (по три потока на одно

ядро), с учетом того, что в случае использовании одного потока на Xeon Phi такой же временной показатель составлял около 643,4 секунды.

На рис. 3 представлены результаты сравнения времен счета при различном соотношении числа MPI-процессов и OpenMP-потоков для различных вычислительных устройств узла в различных режимах.

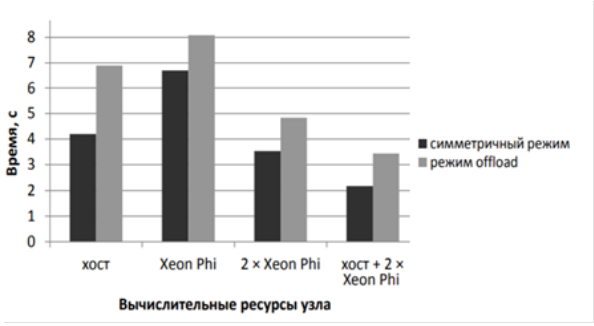


Рис. 3 Времена счета

C. Совместное использование всех вычислительных ресурсов узла

Целью следующего этапа оптимизации программы является реализация совместного использования всех вычислительных ресурсов одного узла: хост-процессоров и обоих установленных ускорителей Xeon Phi. Существующие средства программирования предоставили два способа достижения этой цели [5]:

• режим «симметричный», в котором различные процессы одной MPI-программы распределяются по различным ядрам хост-процессоров и по доступным ускорителям;
• режим «разгрузочный», в котором в основной программе с помощью директив компилятора выделяются данные и вычисления, которые должны быть выгружены на сопроцессоры.

Для рассматриваемой задачи были реализованы оба режима, и выполнено их сравнение.

На рис. 4 показано сравнение времен выполнения, ускорения и эффективности распараллеливания только при использовании одних хост-процессоров, или всех ресурсов узла в различных режимах в зависимости от задействованного количества узлов кластера. Параметры для распределения потоков и процессов в узле на хост-процессоры в «симметричном» режиме выполнения, а также параметры распределения работ в режиме «разгрузочный» выставлены в соответствии с оптимальными значениями, полученными ранее.

На рис. 5 показано время счета при увеличении размера задачи как пропорция количества узлов, а также эффективность распараллеливания в слабом смысле. На графике видно, что при использовании всех ресурсов узла в симметричном режиме эффективность снижается незначительно, чем при использовании только хост-процессоров, тогда как время выполнения отличается в два раза.

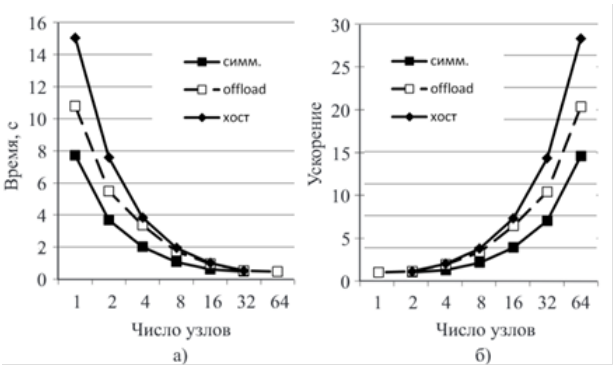


Рис. 4. Время вычисления одного шага по времени (а), ускорение (б) как функция от числа задействованных узлов кластера для сетки размером 1800×1800 при использовании различных ресурсов узла и различных режимов.

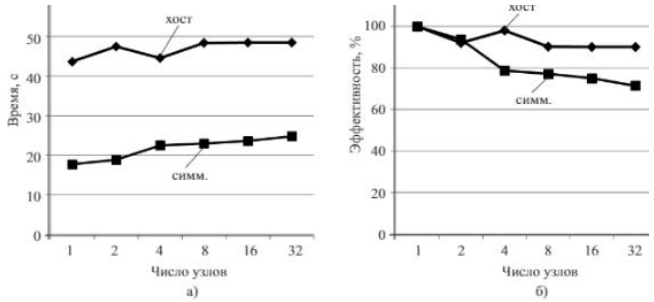


Рис. 5. Время вычисления шага алгоритма по времени (а) и эффективность распараллеливания в слабом смысле (б) как зависимость от числа узлов кластера для вычислительной сетки размером 1800×1800 на узел при использовании различных вычислительных ресурсов узла.

D. Использование нескольких узлов кластера

Разработанный с использованием MPI, OpenMP и «разгрузочных»-директив комплекс программ позволяет производить вычисления на кластере НКС-30Т в симметричном или разгрузочном режиме.

По результатам, при малом числе узлов, соотношение времен выполнения повторяет данное соотношение при одном узле в случае алгоритма Рунге-Кутта. Однако, в случае использовании только процессоров вычислительных систем эффективность методов распараллеливания с увеличением числа узлов снижается с меньшей скоростью, и, начиная с некоторого, так называемого критического числа узлов, задействование сопроцессоров в задачах данного размера станет неэффективным. Дополнительные запуски при других размерностях задачи показывают, что при увеличении размерности данной задачи увеличивается критическое количество узлов.

IV. Заключение

Созданный набор параллельного программного обеспечения, предназначенный для моделирования фильтрационного процесса двухфазной фильтрации жидкости через многослойную среду с порами, был запрограммирован и оптимизирован для использования на производительном вычислительном кластере с установленными многоядерными ускорителями Intel

Xeon Phi 7290 KNL. Были предложены различные способы и методы оптимизации, специфические для данного типа ускорителя, и возможное их влияние на конечное время вычисления программы. Вычислительный комплекс обладает достаточно хорошей масштабируемостью по отношению к числу используемых кластерных узлов. Применение многоядерных ускорителей Xeon Phi 7290 KNL на кластере ССКЦ НКС-30Т позволило снизить время расчета алгоритма по сравнению с вычислениями, использующими только центральные хост-процессоры примерно в два раза.

Методика использования различных режимов работы ускорителей Xeon Phi показывает, что для больших задач самым эффективным является симметричный режим, как с точки зрения программирования, так и с точки зрения достигаемого уровня ускорения производительности.

Библиография

1. Ю.В. Перепечко, Е.И. Роменский, Г.В. Решетова, Н.Л. Перепечко, В.Э. Малышкин, К.В. Калгин, С.Е. Киреев, М.Б. Остапкевич Нелинейная акустика и режимы фильтрации в пористых средах // Суперкомпьютерные технологии в науке, образовании и промышленности: Альманах / Под редакцией академика В.А. Садовничего, академика Г.И. Савина, чл.-корр. РАН Вл.В. Воеводина. – М.: Издательство Московского университета, 2013. С. 119-126.

2. Нелинейный релаксационный механизм генерации шума фильтрации в пористых средах // Известия Вузов. Радиофизика, Лебедев А. В., 2018

3. Об описании течений слабосжимаемой жидкости в пористых средах при нелинейном законе фильтрации // Журнал "Механика жидкости и газа" - ИПМех РАН Н. Е. Леонтьев, 2013

4. Морозов Д. Н., Четверушкин Б. Н., Чурбанова Н. Г., Трапезникова М. А. Моделирование задач фильтрации на гибридных вычислительных системах // Известия ЮФУ. Технические науки. 2012. № 6. С. 87-91.

5. Выполнение программ на Intel Xeon Phi. Модели организации вычислений с использованием Intel Xeon Phi //Режим доступа hpc-education.unn.ru

6. Годунов С.К., Забродин А.В., Иванов М.Я., Крайко А.Н., Прокопов Г.П. Численное решение многомерных задач газовой динамики. М.: Наука. 1976.

Статья получена 7 апреля 2019.

А.С.Первунин – Новосибирский Государственный Университет, РФ (e-mail: a.pervunin@nsu.ru)

Optimization for a cluster with Xeon Phi accelerators for the problem of the filtration flow of liquids in three-dimensional media with double porosity

Andrey Pervunin

Abstract - Based on a previously developed software package for simulating multiphase flows in a deformable porous medium using iterative methods for solving fifth-order Runge-Kutta differential equations, a parallel software package was implemented that was optimized to run on an existing cluster with Intel Xeon Phi accelerators, which are analogues of graphic accelerators in the form of multi-core coprocessors. Although these accelerators have already ceased to be produced, the methods described in this article can provide increased performance on the successors of this processor. Various ways of optimizing the program code described in this article, which are specific to this accelerator, and their impact on the time of the computing program were considered. Comparison of different ways of using accelerators in the cluster was made: native mode, symmetric mode and offload mode. A brief description of these modes was also provided. It was shown that when using the Intel Xeon Phi multi-core coprocessor, it is possible to obtain significant acceleration when performing calculations. Estimates of acceleration and efficiency are obtained using a different number of nodes of a computing cluster when using a software package. The parameters of the target computing cluster are listed.

Keywords: Runge-Kutta method, high-performance computing, scalability, parallel programming, program optimization, Intel Xeon Phi accelerators.

REFERENCES

1. Yu. V. Perepechko, E. I. Romenski, and G. V. Reshetova, “Modeling of Multiphase Flows in FiniteDeformed Porous Media,” in Proc. 11th World Congress on Computational Mechanics (WCCM XI), Barcelona, Spain, July 20–25, 2014 (Polytech. Univ. Catalonia, Barcelona, 2014), pp. 4630–4641.
2. Yu. V. Perepechko, E. I. Romenskii, G. V. Reshetova, et al., “Nonlinear Acoustics and Filtration Regimes in Porous Media,” in Supercomputing Technologies in Science, Education, and Industry, Ed. by V. A. Sadovnichii, G. I. Savin, and Vl. V. Voevodin (Mosk. Gos. Univ., Moscow, 2013), pp. 119–126.
3. E. Saule, K. Kaya, and U. V. ¸Cataly¨urek, “Performance Evaluation of Sparse Matrix Multiplication Kernels on Intel Xeon Phi,” in Lecture Notes in Computer Science (Springer, Heidelberg, 2014), Vol. 8384, pp. 559–570.
4. G. Teodoro, T. Kurc, J. Kong, et al., “Comparative Performance Analysis of Intel Xeon Phi, GPU, and CPU: A Case Study from Microscopy Image Analysis,” IEEE Trans. Parallel Distrib. Syst. (2014). doi 10.1109/IPDPS.2014.111
5. O. Kaczmarek, C. Schmidt, P. Steinbrecher, and M. Wagner, Conjugate Gradient Solvers on Intel Xeon Phi and NVIDIA GPUs, arXiv preprint: 1411.4439v1 [physics.comp-ph] (Cornell Univ. Library, Ithaca, 2014), http://arxiv.org/abs/1411.4439/. Cited February 7, 2015.
6. S.K Godunov, A.V. Zabrodin, M.Y. Ivanov, A.N.Krajko, G.V. Prokopov Numerical solution of multidimensional problems of gas dynamics. M.: Science. 1976.