Институт информационных и вычислительных технологий МОН РК (Республика Казахстан, г. Алматы)

Институт вычислительной математики и математической геофизики СО РАН (Россия, г. Новосибирск)

Новосибирский государственный университет (Россия, г. Новосибирск)

Институт математики НАН КР (Кыргызская Республика, г. Бишкек)

Институт автоматики и информационных технологий НАН КР (Кыргызская Республика, г. Бишкек)

ОсОО "Силк Роад энд Кыргыз Гайдеде Трипс" (Кыргызская Республика, г. Бишкек)

МАТЕРИАЛЫ
XIV Международной Азиатской школы-семинара «ПРОБЛЕМЫ ОПТИМИЗАЦИИ СЛОЖНЫХ СИСТЕМ»

20 июля – 31 июля 2018 г.

Часть 1

Кыргызская Республика
оз. Иссык-Куль
пансионат «Отель Евразия»

с. Кара-Ой

Алматы
2018

привязки к конкретным вычислительным ресурсам, а также проверять создаваемые параллельные программы с помощью формальных методов.

Применение технологии также повышает эффективность использования супервычислителей за счет переноса работ программистов по конструированию и отладке программ с дорогих супервычислителей на более дешевые и привычные персональные компьютеры, а также за счет снятия необходимости прикладному программисту выполнять построение, верификацию и отладку программы для решения одной и той же задачи каждый раз заново при переходе с одного супервычислителя на другой.

Исследование выполнено за счет гранта Российского научного фонда (проект 18-11-00118).

Список литературы

1. Gaudiot J.-L., DeBoni T., Feo J. X.`, et al. The Sisal project: real world functional programming // Lecture Notices in Computer Science. – 2013. Vol. 1808. – pp. 84–72.
2. Касьянов В. Н., Касьянова Е. В. Язык программирования Cloud Sisal. – Новосибирск, 2018. – 45 с. – (Препринт/ РАН, Сиб. отд-ние, ИСИ; N181).
3. Касьянов В. Н. Аннотирование программ и их преобразование // Программирование. – 1989. № 4. – С. 3–16.
4. Касьянов В. Н. Трансформационный подход к конкретизации программ // Кибернетика. – 1989. № 6. – С. 28–32.
5. Касьянов В. Н. Иерархические графы и графовые модели: вопросы визуальной обработки // Проблемы систем информатики и программирования. – Новосибирск: ИСИ СО РАН, 1999. – С. 7-32.

Касьянов Виктор Николаевич – д.ф.-м.н., профессор,
г.н.с. Института систем информатики им. А.П. Ершова СО РАН,
профессор Новосибирского государственного университета;
630090, Новосибирск; e-mail: kvn@iis.nsk.su

Касьянова Елена Викторовна – к.ф.-м.н., доцент,
с.н.с. Института систем информатики им. А.П. Ершова СО РАН,
доцент Новосибирского государственного университета;
630090, Новосибирск; e-mail: kev@iis.nsk.su

УДК 519.687.1

ПАРАЛЛЕЛЬНЫЙ АЛГОРИТМ РЕШЕНИЯ УРАВНЕНИЯ ПУАССОНА НА ОСНОВЕ ТЕХНОЛОГИИ MPI+OPENMP

Кенжебек Е.Г. [1], Иманкулов Т.С. [1], Ахмед-Заки Д.Ж. [2]

[1] Казахский национальный университет имени аль-Фараби,
[2] Университет международного бизнеса

***Аннотация**. В данной работе рассматривается разработка гибридного параллельного алгоритма с использованием технологий параллельных вычислений MPI и OpenMP. Полученные результаты тестирования алгоритмов представлены и проанализированы, на основании чего описаны преимущества выбранной гибридной архитектуры.*

***Ключевые слова**: высокопроизводительные вычисления, гибридные технологии, параллельные вычисления, MPI, OpenMP.*

Введение

Высокопроизводительные вычисления используют параллельные технологии MPI, OpenMP, CUDA. Используя данные технологии наибольшую производительность можно достичь при создании гибридных алгоритмов с использованием вышеуказанных технологий. Самым высокопроизводительным, при определенном круге задач, будет слияние технологий CUDA, MPI и OpenMP в единое целое. Особый интерес к гибридным алгоритмам вызван тенденцией к использованию для высокопроизводительных вычислений многоядерных архитектур и SMP-кластеров. Одним из наиболее эффективных подходов программирования для таких кластеров является гибридный, основанный на комбинированном использовании MPI и OpenMP. Гибридный подход предполагает, что алгоритм разбивается на параллельные процессы, каждый из которых сам является многопоточным. Таким образом, имеется два уровня параллелизма: параллелизм между MPI процессами и параллелизм внутри MPI процесса на уровне потоков [1].

Существует достаточна много работ, посвященных исследованию MPI / OpenMP подхода. Как показывает практика [2-4], за счет укрупнения MPI процессов и уменьшения их числа гибридная модель может устранить ряд недостатков MPI, таких как большие накладные расходы на передачу сообщений и слабая масштабируемость при увеличении числа процессов. Однако, производительность гибридной технологий очень сильно зависит от режима ее запуска и выполнения, который определяет соотношение числа MPI-процессов и OpenMP-потоков на одном вычислительном узле [5].

Известно, что MPI реализация для алгоритмов, где распределение данных по процессам возникает естественным образом, показывает очень высокую эффективность (почти линейное шкалирование времени от числа MPI процессов). Как было показано, например, в [6], для достижения сравнимой производительности на одном вычислительном узле в случае OpenMP реализации требуется реализация OpenMP с использованием концепции, на которой основана технология MPI, но с учетом наличия общей памяти на узле.

1. Постановка задачи и метод решения

Целью данной работы является создание гибридной программы, которая решает двухмерное уравнение Пуассона с помощью итерационного метода Якоби, на языке программирования C++ с использованием технологий MPI и OpenMP.

Двухмерное уравнение Пуассона вида:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = -f(x, y)$$

где u(x,y) – искомая функция; f(x,y) – некоторая непрерывная функция, на прямоугольной области с граничными условиями Дирихле.

$$f(x,y) = 2x(1-x) + 2y(1-y)$$

Граничные условия Дирихле для рассматриваемой задачи:

$$u(0,y) = 0;$$
$$u(1,y) = 0;$$
$$u(x,0) = 0;$$
$$u(x,1) = 0;$$

Наиболее распространенный подход для численного решения дифференциальных уравнений является метод конечных разностей. Следуя этому методу, область решения представляется в виде дискретного набора точек [7]. Для проведения дискретизации внутренних точек сетки используется пятиточечный шаблон, таким образом используя метод Якоби для проведения итераций, уравнение обретает следующую форму:

$$u_{i,j}^{n+1} = 0.\,25\left(u_{i+1,j}^{n} + u_{i-1,j}^{n} + u_{i,j+1}^{n} + u_{i,j-1}^{n} + h^2 f_{ij}\right)$$

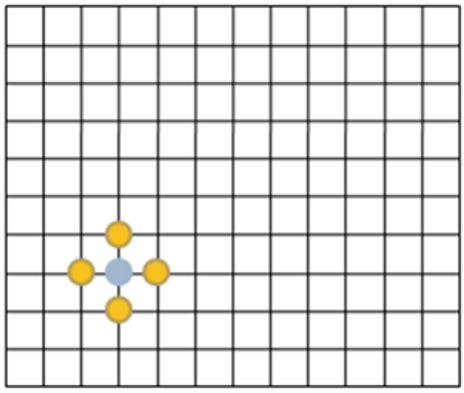

Рис. 1. Метод Якоби

Здесь $u_{i,j}^{n+1}$ является новым временным слоем итераций Якоби, а $u_{i,j}^{n}$ предыдущий временной слой итераций. Как показано на рисунке 1, для вычисления значения каждой точки нового слоя $u_{i,j}^{n+1}$, понадобятся значения четырех соседних точек предыдущего слоя $u_{i+1,j}^{n}, u_{i-1,j}^{n}, u_{i,j+1}^{n}, u_{i,j-1}^{n}$.

2. Параллельный алгоритм решения задачи с помощью MPI

Первое что нужно решить при распараллеливании подобных задач это – способ разделение данных между вычислительными узлами.

Для решения данной задачи использовалась ленточная схема разделение данных. С подобным разделением данных, расчетную область можно разбить на несколько горизонтальных полос. На каждый процесс, который выполняет обработку какой-либо полосы, были продублированы граничные строки предыдущей и следующей полосы. Полученные расширенные полосы показаны на рисунке 2 пунктирными рамками. Расчеты в каждой полосе выполняются независимо друг от друга и перед каждой новой итераций Якоби необходимо обновлять продублированные граничные строки.

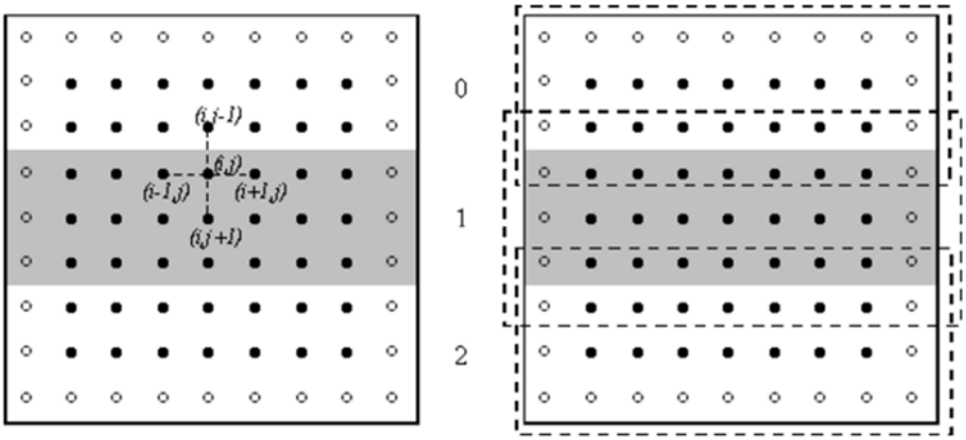

Рис. 2. Декомпозиция области

Обмен граничных строк между процессами состоит из двух пересылок данных. Первое, каждый процесс передает свою нижнюю граничную строку следующему процессу и принимает верхнюю граничную строку того процесса. Во втором случае, передача граничных строк выполняется в обратном направлении, то есть каждый процесс передает свою верхнюю граничную строку предыдущему процессу и принимает нижнюю граничную строку от того процесса. Для этой операций использовалось совмещенные прием и передача сообщений MPI_SendRecv.

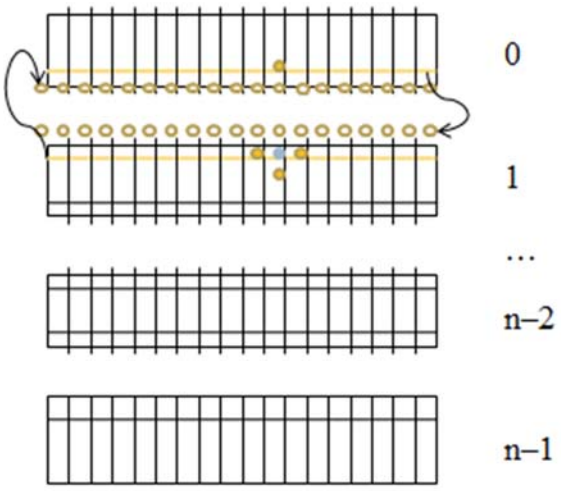

Рис. 3. Обмен граничных строк между процессами

3. Параллельный алгоритм решения задачи с помощью OpenMP

При организации рассматриваемой задачи по технологию OpenMP (Open Multi-Processing) в последовательный программный код были добавлены директивы компилятора. В рамках данной технологии эти директивы используются для выделения нескольких параллельных областей, в которых обработка выполняется с помощью потоков, так называемых threads. Используемые процессоры являются многоядерными, чтобы оптимально загрузить работой все ядра, в программе должно быть несколько параллельных потоков. Количество потоков которые задаются в программе не должно превышать число ядер [8].

Программный код был получен из исходного последовательного кода путем добавления директив и обращения к библиотеке OpenMP. Количество потоков используемых в программе устанавливалось с помощью функции omp_set_num_threads (). Параллельные области в данной программе заданы с помощью директив **parallel for**. Компилятор, который поддерживает технологию OpenMP, разделяет выполнение итераций цикла между потоками программы. Параметр директивы **private** опеределяет доступность данных в потоках программы, то есть для переменных с описанием **private** создаются отдельные копии для каждого потока. Тем самым, эти параллельные потоки работают независимо друг от друга. С помощью директивы **reduction** в каждом потоке создаются локальные копии переменных. Используя операцию нахождения максимума (max) среди этих локальных перемен dmax определяется максимум. Таким образом, достигнута параллельность работы в потоках по технологию OpenMP.

4. Гибридный параллельный алгоритм

После создания параллельных алгоритмов MPI и OpenMP был разработан гибридный метод MPI+OpenMP для параллельного выполнения вычислений. При создании гибридной программы была рассмотрена подходящая архитектура для решения рассматриваемой задачи и принята во внимания преимущества каждой из технологий.

Технология MPI используется для распараллеливания задачи между SMP узлами на процессы, что позволяет использовать адресные пространства и вычислительные ресурсы процессоров. При выполнений расчета на каждом узле не используется все преимущества разделяемой между ядрами процессора памяти, поэтому технология OpenMP служит для распараллеливания между ядрами каждого из этих SMP узлов. MPI запускался в кластерной конфигураций, который использует вычислительные ресурсы нескольких процессоров и запускает несколько отдельных MPI процесса на каждом используемом узле [9].

Распределение данных между процессами выполнялось с помощью MPI, а параллельный расчет данных внутри каждого из процессов обрабатывалось OMP потоками.

Использованная архитектура показана на рисунке 4. Такая архитектура имеет несколько режимов работы с MPI процессами и OMP потоками. В гибридной технологии произведение MPI процессов и потоков (threads) задающиеся в программе должно совпадать с общим количеством используемых ядер в вычислительной системе. Это используется для того чтобы правильно загрузить работой все ядра. Если превысить определенное количество потоков задающихся на каждый MPI процесс, это может привести к столкновению потоков между собой. Тем самым, это приведет к увеличению времени выполнения работы.

Как показана на рисунке 4, например, если нам доступны 2 восьмиядерных узла, то произведение используемых процессов и потоков не должно превышать число 16. Это обеспечит балансировку работы между процессами и потоками.

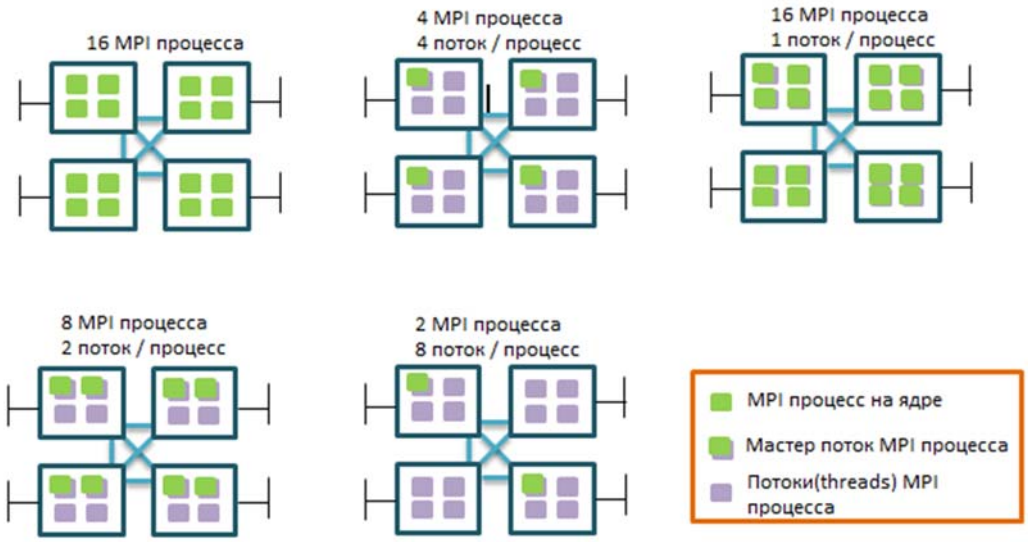

Рис. 4. Архитектура гибридной программы MPI+OpenMP

5. Результаты

Все параллельные программы были протестированы на кластере НГУ, у которой было доступно 2 узла. На каждом узле имеется по два 4х ядерного процессора Intel(R) Xeon(R) CPU E5-2603 v2 1.80GHz.

На основании полученных данных были расчитаны полученное среднее ускорение и средняя эффективность параллельных программ MPI и MPI+OpenMP для решения уравнения Пуассона. Результаты приведены в графиках 8 и 9.

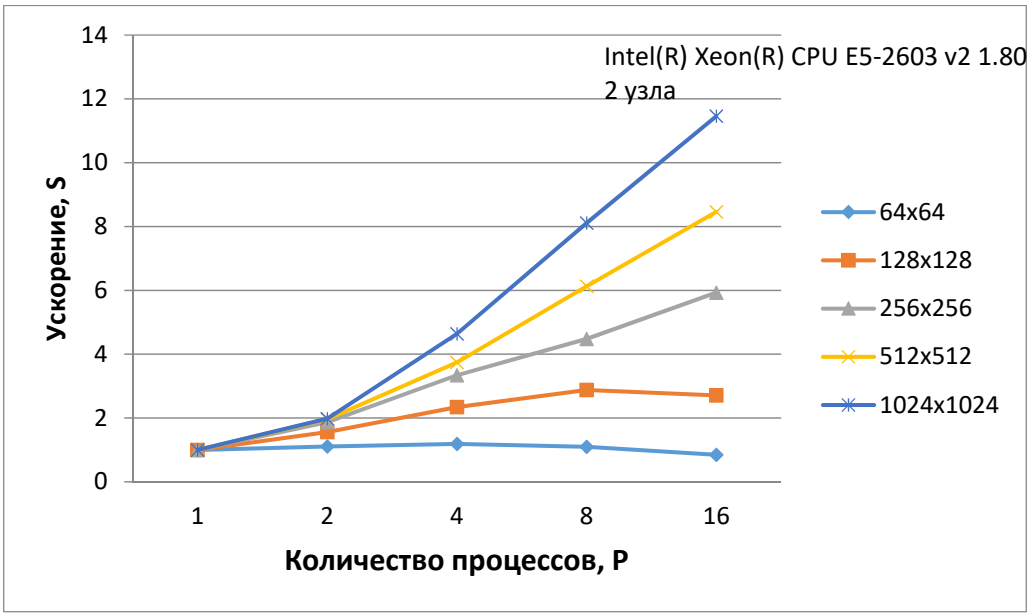

Рис. 5. Полученное ускорение для параллельной версий программы на MPI

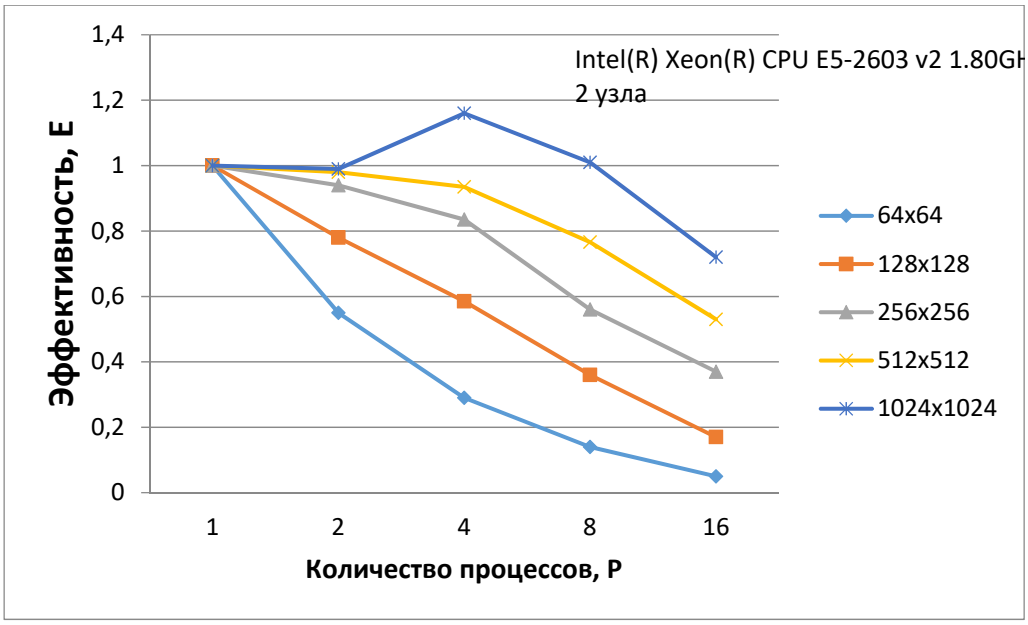

Рис. 6. Полученная эффективность параллельной версий программы на MPI

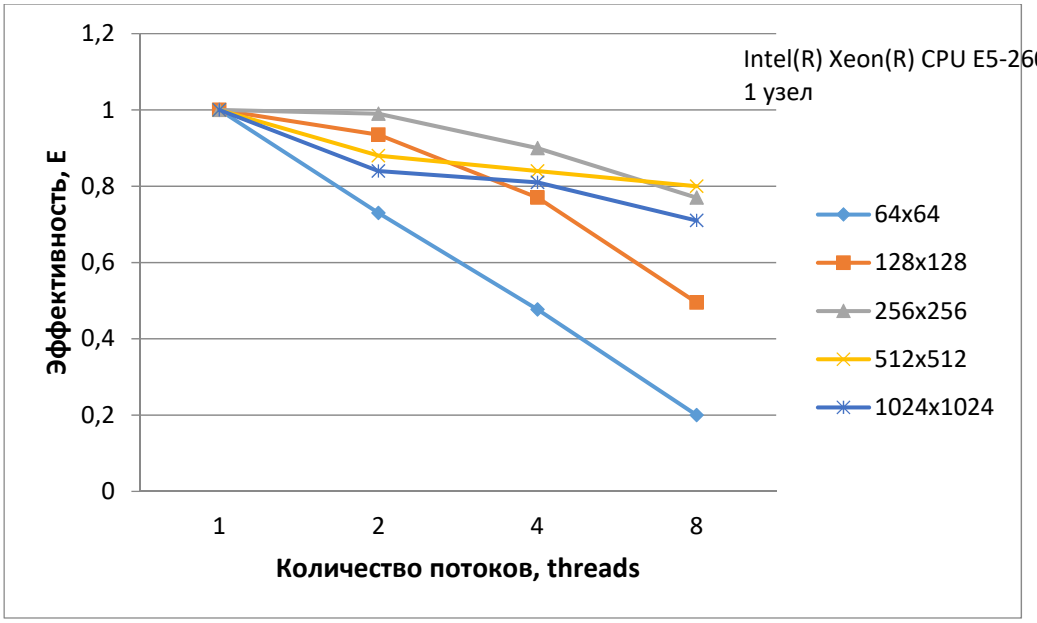

Рис. 7. Полученная эффективность для параллельных версий программы на OpenMP

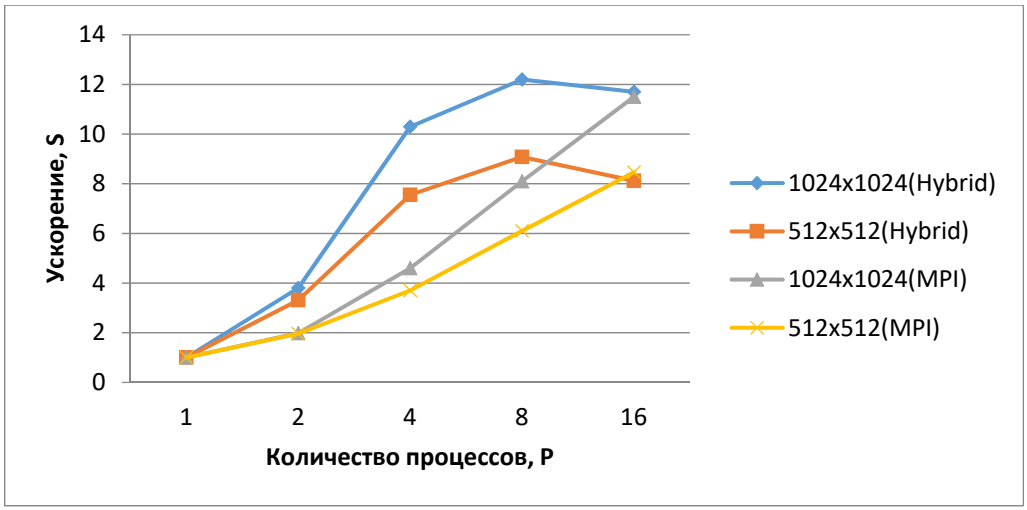

Рис. 8. Полученное ускорение для параллельных версий программ

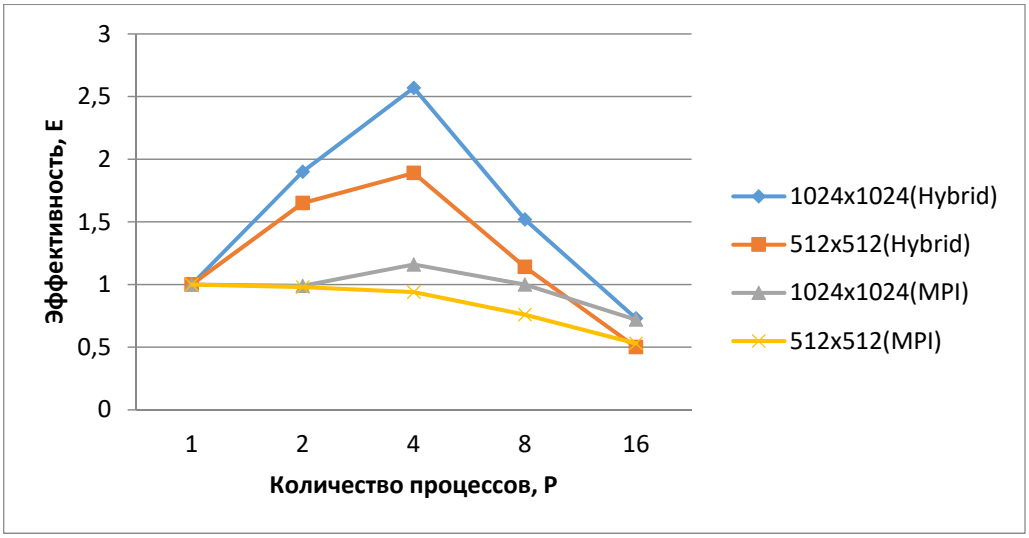

Рис. 9. Полученная эффективность параллельных версий программ

6. Заключение

Данная работа была посвящена разработке гибридной программы с использованием технологии MPI и OpenMP. Гибридная реализация программы с использованием технологии MPI и OpenMP более эффективна при работе на большом количестве узлов и при использовании многоядерных процессоров, так как данная гибридная технология имеет возможность использовать ядра процессоров нескольких вычислительных узлов. Гибридная программа MPI+OpenMP для решения уравнения Пуассона ускорила производительность работы в 1.5-2 раза в сравнении с MPI программой.

Список литературы

1. Горобец А.В., Суков С.А., Железняков А.О. Расширение двухуровневого распараллеливания MPI+OpenMP посредством OpenCL для газодинамических расчетов на гетерегенных системах // Вестник ЮУрГУ. - 2009. - № 9. - С. 76-86.

2. Makris I. Mixed Mode Programming on Clustered SMP systems // Thesis in high performance computing. The University of Edinburgh, 2005. –108 p.
3. Rane A., Stanzione D. Experiences in tuning performance of hybrid MPI/OpenMP applications on quad-core systems. Proceedings 10th LCI International Conference on High-Performance Clustered Computing. – 2009. – P. 1-10.
4. Rabenseifner R., Hager G., Jost G. Hybrid MPI/OpenMP parallel programming on clusters of multi-core SMP nodes // Proc. 17 Euromicro Internat. Conf. on Parallel, Distributed and Network-based Processing. Weimar, 2009. P. 427–436.
5. Крюков А.П., Степанова М.М. Эффективный запуск гибридных параллельных задач // Вестник ЮУрГУ. – 2013. - № 3. – С. 32-48.
6. Mitin I., Kalinkin A., Laevsky Yu. A parallel iterative solver for positive-definite systems with hybrid MPI-OpenMP parallelization for multi-core clusters // J. Comput. Sci. - 2012. - V. 6, № 3. - P. 463–468.
7. Рындин Е. А. Методы решения задач математической физики. – Таганрог: Изд-во ТРТУ, 2003. -119 с.
8. Антонов А. С. Параллельное программирование с использованием технологии OpenMP. -М. : Изд-во МГУ, 2009. -77 с.
9. Ахмед-Заки Д.Ж., Борисенко М.Б. Разработка высокопроизводительных приложений с использованием гибридных технологий параллельных вычислений - MPI/OpenMP/Cuda. – Алматы: НИИ Институт КазНУ, 2013. -7 с.

Работа выполнена в рамках грантового финансирования Комитета Науки МОН РК.

Ержан Галымжанович Кенжебек – магистрант КазНУ имени аль-Фараби; 050000, Алматы, Казахстан; erzankenzhebek@gmail.com

Тимур Сакенович Иманкулов – PhD, и.о. доцента КазНУ имени аль-Фараби; 050000, Алматы, Казахстан; e-mail: imankulov.timur@gmail.com

Дархан Жумаканович Ахмед-Заки – д.т.н., профессор Университет международного бизнеса; 050000, Алматы, Казахстан; e-mail: darhan_a@mail.ru

ПРИМЕНЕНИЕ ЯКОБИАНА В РЕШЕНИИ ГИПОТЕЗЫ РИМАНА О НЕТРИВИАЛЬНЫХ НУЛЯХ ДЗЕТА ФУНКЦИИ

Керимбаев Рашид Конырбаевич

Казахский Национальный Университет имени Аль-Фараби

Аннотация. *Рассматривается известная проблема Римана о нетривиальных нулях дзета функции*

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}, \qquad (1)$$

где s= r+it – комплексное число.

Тривиальные нули, а также некоторые нетривиальные нули дзета функции (1) известны.