Министерство образования и науки Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»

Кафедра математического обеспечения вычислительных
комплексов и информационных систем

СРАВНЕНИЕ ЭФФЕКТИВНОСТИ РАСПАРАЛЛЕЛИВАНИЯ МЕТОДА КОНЕЧНЫХ ЭЛЕМЕНТОВ НА АРХИТЕКТУРЕ X86_64 И INTEL XEON PHI

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 441 группы
направления 02.03.03 Математическое обеспечение и администрирование информационных систем
факультета компьютерных наук и информационных технологий
Егорова Максима Алексеевича

Научный руководитель
профессор, д.ф.-м.н. _____ Андрейченко Д. К.
подпись, дата

Зав. кафедрой _____ Андрейченко Д. К.
подпись, дата

Саратов 2016

ВВЕДЕНИЕ

Актуальность темы. В настоящее время наблюдается повышенный интерес к распараллеливанию вычислений при решении задач математической физики на современных высокопроизводительных вычислительных системах, в частности, при конечно-элементном моделировании задач математической физики. Вместе с тем, не изучен полностью вопрос о возможности распараллеливания типовых вычислительных операций, выполняемых в процессе работы современных программных комплексов конечно-элементного моделирования.

Конечно не любая задача может быть эффективна перенесена на многоядерный процессор, однако существует класс задач, для которых такой подход дает существенный выигрыш в скорости. Например, хорошо распараллеливаются задачи линейной алгебры с плотно заполненными матрицами достаточно большой размерности.

Хорошо известно, что именно операция факторизации разреженных матриц большой размерности является наиболее трудоемкой типовой операцией конечно-элементного моделирования [1]. В частности, эффективность факторизации разреженных симметричных положительно определенных матриц определяет эффективность решения задачи спектра собственных частот и форм колебаний объектов управления с распределенными по пространству параметрами, моделируемых на основе аппарата краевых задач для уравнений в частных производных. Распараллеливание данной типовой операции на основе многопоточности на симметричных мультипроцессорных системах с общей памятью реализовано в свободно распространяемом пакете CHOLMOD и в коммерческих библиотеках Intel Math Kernel Library, а распараллеливание на основе технологии MPI на кластерных системах с распределенной памятью – в свободно распространяемом пакете MUMPS и в старших версиях Intel Math Kernel Library.

Целями данной работы являются сравнение эффективности распараллеливания типовых конечно-элементных операций, оптимизированных на основе OpenMP, с использованием компиляторов Intel и математических функций библиотек поддержки высокопроизводительных вычислений Intel MKL, а также оптимизированных на основе OpenMP с использованием компиляторов Intel + Intel MKL для работы на сопроцессоре-ускорителе Intel Xeon Phi.

Задачи данной работы:

1) Сборка базовой версии конечно-элементного решателя ELMER для ОС Linux x86_64 с использованием компиляторов GNU.

2) Сборка оптимизированной на основе OpenMP версии конечно-элементного решателя ELMER для ОС Linux x86_64 с использованием компиляторов Intel и математических функций библиотек поддержки высокопроизводительных вычислений Intel MKL.

3) Сборка оптимизированной на основе OpenMP версии конечно-элементного решателя ELMER для архитектуры Intel MIC и предназначенной для работы на сопроцессоре-ускорителе Intel Xeon Phi с использованием компиляторов Intel и библиотек Intel MKL.

4) Подготовка конечно-элементных сеток в генераторе конечно-элементных сеток GMSH.

5) Подготовка проекта и файла с заданием для конечно-элементного решателя ELMER для запуска на узле кластера и на сопроцессоре-ускорителе Intel Xeon Phi

6) Исследование эффективности базовой и оптимизированных версий конечно-элементного решателя ELMER

Структура и объём работы. Дипломная работа состоит из введения, семи разделов, заключения, списка использованных источников и приложения.

Общий объем работы – 47 страниц, из них 42 страницы – основное содержание, включая 11 рисунков и 4 таблицы, список использованных источников – 20 наименований. Все основные научные результаты, программные реализации и выводы, изложенные в дипломной работе, получены самостоятельно.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Первый раздел «Основы конечно-элементного моделирования». В этом разделе в первом подразделе были рассмотрены основные уравнения теории упругости [2]; рассмотрены внутренние напряжение; описаны малые колебания упруго деформируемого тела; рассмотрен так называемый упругий потенциал; поставлены начальные и граничные условия для уравнений в частных призводных. Во втором подразделе рассмотрен проекционный метод Галёркина [3]; показаны основные методы и отличительные особенности конечных элементов [4, 5]; рассмотрена факторизация разреженных матриц, матрица инерции, матрица жесткости, система обыкновенных дифференциальных уравнений [6]. В третьем подразделе рассмотрены собственные частоты и частичная проблема собственных значений; исследование собственных частот и форм колебаний упруго деформируемых тел на основе метода конечных элементов сведено к численному решению обобщенной задачи на собственные значения [7]; рассмотрены классические численные методы и решение численных задач на собственные значения [8].

Второй раздел «Библиотеки поддержки высокопроизводительных вычислений, обеспечивающие эффективность конечно-элементного моделирования». В этом разделе говорится об эффективности конечно-элементного моделирования и его зависимости от библиотек поддержки высокопроизводительных вычислений BLAS и LAPACK [9]. Так же в данном разделе говорится о факторизация разреженных матриц и какие алгоритмы используются. При решении систем линейных уравнений с несимметричными разреженными матицами используется свободно распространяемый пакет UMFPACK [10]. Подробности решения систем линейных уравнений с разреженными матрицами показано в работе Джордж А., Лю Дж [11].

Третий раздел «Модель программирования OpenMP». В данном разделе говорится об популярности использования технологии параллельного программирования OpenMP и об ее стандартах [12, 13]. Технология OpenMP

нацелена на то, чтобы пользователь имел один вариант программы для параллельного и последовательного выполнения. OpenMP реализует параллельные вычисления с помощью многопоточности, в которой «главный» (master) поток создает набор «подчиненных» (slave) потоков, и задача распределяется между ними. В этом разделе рассмотрен POSIX-интерфейс для организации нитей (Pthreads). Важным достоинством технологии OpenMP является возможность реализации так называемого инкрементального программирования, когда программист постепенно находит участки в программе, содержащие ресурс параллелизма, с помощью предоставляемых механизмов делает их параллельными, а затем переходит к анализу следующих участков. Так же в данном разделе рассмотрена компиляция программы, модель параллельной программы, директивы и функции и само выполнение программы.

Четвертый раздел «Архитектура Intel MIC и сопроцессоры Intel Xeon Phi». В четвертом разделе говорится о возможностях сопроцессоров-«ускорителей» Intel Xeon Phi 5110P и архитектуры Intel MIC [14]. Говорится о SIMD-инструкциях, которые выполняются над векторными 512-битными операндами, для хранения которых в каждом ядре могут использоваться 32 новых 512-битных регистров. Таким образом, можно выполнить векторные операции над наборами восьми 64-битных вещественных чисел одновременно. Повышенное количество SIMD-регистров позволяет сопоставить каждому ядру Intel MIC до четырех логических потоков одновременно. Главным преимуществом Intel MIC является большая гибкость его использования, чем ускорителей ATI и Nvidia. Ускорители Xeon Phi обеспечивают ускорение, сопоставимое с акселераторами Nvidia Tesla. Наилучшие результаты достигаются именно при работе программ, оптимизированных для Intel MIC. Наличие версии ОС Linux, предназначенной для работы на архитектуре Intel MIC как на мультиядерном центральном прорцессоре, обеспечивает возможность использования ускорителей Intel Xeon Phi как узлов компьютерного кластера, и именно в таком варианте наблюдается наибольшая вычислительная

эффективность. Генерация объектных файлов и загрузочных образов для архитектуры Intel MIC осуществляется при помощи компиляторов Intel C/C++/Fortran (icc/icpc/ifort) 13.1 для ОС Linux x86_64 в режиме кросс-компиляции. Генерация многопоточного кода для архитектуры Intel MIC поддерживается стандартными директивами OpenMP.

Пятый раздел «Система конфигурирования и сборки проектов CMake». В данном разделе говорится об истории создания CMake и предъявленные к нему требования. Система CMake должна была иметь улучшенный способ конфигурирования, сборки и развертывания сложного программного обеспечения, предназначенный для большого количества различных платформ. Система должна была быть простой в использовании и должна была помочь наиболее продуктивно использовать время, затрачиваемое исследователями на программирование. Было решено, что новая система сборки должна быть разработана для ITK и, в общем случае, для C++. Так же выделены основные требования к новой системе сборки. В данном разделе рассмотрена реализация CMake [15]. Процедура использования CMake состоит из двух основных этапов. Во-первых, это шаг "конфигурирования", на котором пакет CMake обрабатывает все переданные ему входные данные и создает внутреннее представление сборки, которую нужно выполнить. Затем идет следующий этап - шаг "генерации". На этом этапе создаются файлы фактической сборки.

Шестой раздел «Основные возможности, конфигурирование и сборка конечно-элементного решателя ELMER». В подразделе «основные возможности конечно-элементного решателя ELMER» описано множестве базовые математических моделях (в моем случаи используется теория упругости) [16]. При этом имеется возможность численного моделирования развития физических процессов во времени в одномерных, двумерных и трехмерных пространственных областях сложной формы [17]. В следующем подразделе, для дальнейшего проведения вычислительных экспериментов, были собраны три версии конечно-элементного решателя ELMER (базовая и две версии

оптимизированные на основе OpenMP, одна из которых для работы на сопроцессоре-ускорителе Intel Xeon Phi). Далее поэтапно рассмотрен каждый шаг генерации конечно-элементных сеток GMSH [18]. Были указаны все необходимые параметры и сгенерирована сетка. В следующем подразделе поэтапно показана подготовка задания для запуска конечно-элементного решателя в ElmerGUI. Конвертирование конечно-элементной сетки из формата хранения GMSH в формат хранения конечно-элементного решателя Elmer (с учетом возможного удаления фиктивных элементов и необходимости перевода миллиметров в метры) реализуется при помощи запуска следующего программного компонента пакета Elmer: Elmergrid 14 2 <файл-с-сеткой>.msh –autoclean –scale 0.001 0.001 0.001 [19]. Дальнейшее создание проекта может быть выполнено в графической оболочке ElmerGUI [20]. Далее сам файл с расчетным заданием для конечно-элементного решателя и конечно-элементные сетки передаются на кластер, где и реализуется запуск собственно конечно-элементного решателя.

Седьмой раздел «Исследование эффективности распараллеливания на основе многопоточности». В таблице 1 приведены первые 5 частот собственных колебаний упругой конструкции, найденные при помощи конечно-элементного решателя ELMER, с использованием ранее сгенерированных конечно-элементных сеток.

Таблица 1. – частоты собственных колебаний упругой конструкции.

Сетка	Собственные частоты, *Гц*				
Тетраэдр 1го прядка	1202,048	1203,94	1433,874	1444,085	2752,96
Полный тетраэдр 2го прядка	1115,206	1117,102	1318,792	1324,827	2632,919
Неполный тетраэдр 2го прядка	1116,722	1118,588	1320,591	1326,697	2637,325

Данные, представленные в таблице 1, подтверждают стабильную работу конечно-элементного решателя ELMER.

В таблице 2 приведено характерное время решения задачи о нахождении первых пяти собственных частот колебания упругой конструкции. Для сетки из тетраэдров первого порядка содержащий примерно 167 тыс. степеней свободы. Результаты приведены для базовой версии ELMER собранной при помощи GNU C++, многопоточной версии с поддержкой функциональности Intel MKL собранный при помощи компилятора Intel для x86_64 и многопоточный версии с поддержкой функциональности Intel MKL для архитектуры Intel MIC собранный при помощи компиляторов Intel для Intel MIC. В исходной версии пакета ELMER факторизация разряженных матриц выполнялась на основе пакета UMFPACK. В оптимизированных версиях собранных при помощи компиляторов Intel факторизация разряженных матриц выполнялась на основе реализации PARDISO из состава библиотек Intel MKL

Таблица 2. - время решения задачи (в секундах) для сетки содержащий 167 тыс. с. с.

Тетраэдр 1го прядка	Тест№1	Тест№2	Тест№3	Тест№4	Тест№5	Среднее, м.
GNU	242	244	243	243	245	4:04
Intelx86_64+MKL	46	46	47	48	46	0:46
IntelMIC + MKL	110	115	119	107	113	1:53

Таблица 3. - время решения задачи (в секундах) для сетки содержащий 175 тыс. с. с.

Полный тетраэдр 2го прядка	Тест№1	Тест№2	Тест№3	Тест№4	Тест№5	Среднее, м.
GNU	281	280	281	281	281	4:41
Intelx86_64+MKL	52	53	53	52	53	0:53
Intel MIC + MKL	131	135	127	130	136	2:12

Таблица 4. - время решения задачи (в секундах) для сетки содержащий 125 тыс. с. с.

Неполный тетраэдр 2го прядка	Тест№1	Тест№2	Тест№3	Тест№4	Тест№5	Среднее, м.
GNU	123	122	122	123	122	2:03
Intelx86_64+MKL	25	24	24	24	25	0:24
Intel MIC + MKL	62	61	65	60	61	1:02

В таблице 3 и 4 приведены аналогичные данные для конечно элементных сеток для полных и не полных тетраэдров второго порядка, содержащих примерно 175 тыс. и 125 тыс. степеней свободы соответственно. Как видно переход к многопоточной оптимизированной версии значительно сокращает характерное время компьютерного моделирования.

ЗАКЛЮЧЕНИЕ

Представленные в таблицах 2, 3 и 4 результаты измерения времени учитывают характерное время всех типовых операций конечно-элементного моделирования: вычисление элементов конечно-элементных матриц, факторизация разреженных матриц, решение систем линейных уравнений после факторизации матриц и т.д. Переход к многопоточной версии позволяет значительно сократить характерное время конечно-элементного моделирования на двух четырехъядерных центральных процессорах Intel Xeon.

Вместе с тем, для эффективного использования сопроцессоров ускорителей Intel Xeon Phi требуется значительный ресурс параллелизма, когда внутри сопроцессора решаемая задача выполняется примерно в 200 потоков OpenMP. Поскольку характерное время конечно элементного моделирования на Intel Xeon Phi примерно в 2,5 раза больше чем характерное время решения данной задачи на двух четырехъядерных центральных процессорах Intel Xeon то можно сделать вывод что при данной размерности решаемая задача не обладает ресурсом параллелизма для эффективного использования Intel Xeon Phi. Возможно, что более трудоемкие задачи конечно-элементного моделирования с более подробными и содержащими большее число узлов конечно-элементными сетками будут лучше реализованы на Intel Xeon Phi. Однако следует учитывать некоторое ограничение на объем внутренней памяти Intel Xeon Phi (как правило 8 ГБ.)

Проведенные исследования позволяют сделать следующие выводы:

1. Базовая версия конечно-элементного решателя ELMER может быть сконфигурирована и собрана для любой ОС, для которой имеется комплект свободно распространяемых компиляторов GNU C++/Fortran и свободно распространяемая утилита конфигурирования и сборки программных проектов CMAKE.
2. При наличии компиляторов Intel C++/Fortran, например, для ОС Linux x86_64, имеется возможность конфигурирования и сборки

оптимизированной версии конечно-элементного решателя ELMER с поддержкой распараллеливания на основе OpenMP и дополнительных математических функций из состава библиотек поддержки высокопроизводительных вычислений Intel MKL.

3. Конечно-элементный решатель ELMER может быть адаптирован для конфигурирования и дальнейшей сборки оптимизированной версии для архитектуры Intel MIC, предназначенной для работы на сопроцессрах-ускорителях Intel Xeon Phi.

4. Переход к многопоточной версии позволяет значительно сократить характерное время конечно-элементного моделирования на двух четырехъядерных центральных процессорах Intel Xeon.

5. Поскольку характерное время конечно элементного моделирования на Intel Xeon Phi примерно в 2,5 раза больше чем характерное время решения данной задачи на двух четырехъядерных центральных процессорах Intel Xeon то можно сделать вывод что при данной размерности решаемая задача не обладает ресурсом параллелизма для эффективного использования Intel Xeon Phi.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Андрейченко Д.К., Ирматов П.В., Ирматова М.С., Щербаков М.Г. О реализации конечно-элементного моделирования в задачах остеосинтеза на кластерных системах СГУ – Изв. Сарат. ун-та. Нов. сер. 2010. Т. 10. Сер. Математика. Механика. Информатика. Вып. 3. С. 77-85.
2. Лурье А.И. Теория упругости. – М.: Наука, 1987. – 368с.
3. Флетчер К. Численные методы на основе метода Галеркина. – М.: Мир, 1988. – 352с.
4. Галлагер Р. Метод конечных элементов. Основы. – М.: Мир, 1984. – 428с.
5. Зенкевич О. Метод конечных элементов в технике. – М.: Мир, 1975. – 541с.
6. Андрейченко Д.К., Велиев В.М., Ерофтиев А.А., Портенко М.С. Теоретические основы параллельного программирования. Электр. пособие. – Саратовский госуниверситет им. Н.Г. Чернышевского. 2015. – http://library.sgu.ru/uch_lit/1255.pdf
7. Калиткин Н. Н. Численные методы. – М.: Наука, 2005. – 224 с.
8. Lehouck R.B., Sorensen D.C., Yang C. ARPACK Users' Guide: Solution of Large Scale Eigenvalue Problems with Implicitly Restarted Arnoldi methods. – www.caam.rise.edu – 1997. – 140 p.
9. Intel Math Kernel Library for the Windows OS User Guide. – Intel, Document Number: 315930-006US (http://developer.intel.com). – 2008. – 135 с.
10. Davis T.A. UMFPACK v.5.7.6 User Guide. – Dept. of Computer and Information Science and Engineering Univ. of Florida (http://faculty.cse.tamu.edu/davis/SuiteSparse/). – 2016. – 139 с (дата обращения 27.05.2016). Яз.англ.
11. Джордж А., Лю Дж. Численное решение больших разреженных систем уравнений. – М.: Мир, 1984. – 333 с.
12. OpenMP Application Program Interface. Version 4.5 - November 2015. [Электронный ресурс]/ OpenMP Architecture Rewiew Board. Электрон. дан. 2015. Режим доступа: http://www.openmp.org/mp-documents/openmp-4.5.pdf, свободный. Загл. с экрана (дата обращения 31.01.2016). Яз.англ.
13. Гергель В.П. Высокопроизводительные вычисления для многопроцессорных многоядерных систем. – М.: Изд-во Моск. ун-та, 2010. – 421 с.
14. Intel® Xeon Phi™ Coprocessor (codename Knights Corner) [Электронный ресурс]/ Intel. – Электрон. дан. – 2013. – Режим доступа:

http://software.intel.com/en-us/articles/intel-xeon-phi-coprocessor-codename-knights-corner, свободный – Загл. с экрана (дата обращения 31.01.2016). Яз.англ.
15. Reference manuals [Электронный ресурс] https://cmake.org/cmake/help/v3.6/#reference-manuals (дата обращения 31.01.2016).
16. Elmer Models Manual, Peter Raback, Mika Malinen, Juha Ruokolainen, Antti Pursula, Thomas Zwinger, Eds, CSC – IT Center for Science, March 30, 2016 [Электронный ресурс] http://www.nic.funet.fi/pub/sci/physics/elmer/doc/ElmerModelsManual.pdf
17. Elmer Solver Manual. – CSC – IT Center for Science (http://www.nic.funet.fi/pub/sci/physics/elmer/doc/ElmerSolverManual.pdf). – 2016. – 107 с (дата обращения 27.05.2016). Яз.англ.
18. Gmsh Reference Manual. The documentation for Gmsh 2.12. A finite element mesh generator with built-in pre- and post-processing facilities 6 March 2016 [Электронный ресурс] http://gmsh.info/doc/texinfo/gmsh.pdf
19. ElmerGrid Manual, Peter Raback, CSC – IT Center for Science, Oct 30, 2015 [Электронный ресурс] http://www.nic.funet.fi/pub/sci/physics/elmer/doc/ElmerGridManual.pdf
20. Elmer GUI Tutorials, CSC – IT Center for Science, March 15, 2016 [Электронный ресурс] http://www.nic.funet.fi/pub/sci/physics/elmer/doc/ElmerTutorials.pdf