ОПТИМИЗАЦИЯ И ПРИМЕНЕНИЕ ПАКЕТА MUMPS ДЛЯ РЕШЕНИЯ ТРЕХМЕРНЫХ СТАЦИОНАРНЫХ ЗАДАЧ ПРОЧНОСТИ НА КЛАСТЕРНЫХ СИСТЕМАХ

С.А. Лебедев, И.Б. Мееров, А.В. Сысоев, Ю.Г. Бартенев, С.И. Бастраков, Е.А. Козинов, И.Г. Лебедев, А.Ю. Малова, А.Н. Стаканов, Н.В. Старостин

Предложен подход к оптимизации MUMPS – одного из наиболее известных прямых решателей СЛАУ, распространяемого в исходных кодах. Основная идея подхода – учет блочной структуры матриц в разреженных СЛАУ, возникающих при решении трехмерных стационарных задач прочности методом конечных элементов. Внедренные на этапе переупорядочивания матрицы улучшения позволили сократить общее время решения задач на ~20% при использовании меньшего числа вычислительных устройств. Рассмотрен вопрос практического применения решателя с учетом конкретного программно-аппаратного окружения. Показана существенная зависимость времени решения задачи от режима запуска (использование разного числа процессов/потоков). Результаты получены на кластере РФЯЦ ВНИИЭФ при решении разреженных СЛАУ с симметричными положительными определенными матрицами.

Введение

Одной из актуальных задач алгебры разреженных матриц является решение систем линейных алгебраических уравнений (СЛАУ) Ax = b с разреженной симметричной положительно определенной матрицей A. Такие СЛАУ возникают при решении многих научных и инженерных задач из разных предметных областей. Разработка алгоритмов решения таких задач и их высокопроизводительная программная реализация, ориентированная на современные вычислительные архитектуры, представляет большой практический интерес.

На сегодняшний день в мире разработано немало прямых и итерационных решателей СЛАУ, ориентированных на вычислительные системы различной архитектуры (Intel MKL PARDISO, MUMPS, SuperLU, CHOLMOD и другие). Некоторые прямые решатели распространяются в составе инженерных пакетов, таких как, например, ANSYS. Обширные обзоры прямых решателей можно найти по ссылкам [8, 9]. В последнем обзоре присутствуют только свободно распространяемые решатели, поэтому не упоминается один из наиболее быстрых пакетов для систем с общей памятью – Intel MKL PARDISO [2, 10].

В данной работе рассматривается MUMPS [11] – один из лучших прямых решателей, работающий на кластерных системах традиционной архитектуры, распространяемый в исходных кодах и допускающий использование сторонних высокопроизводительных библиотек. Основное внимание уделяется вопросам оптимизации и выбора режима запуска MUMPS при решении трехмерных стационарных задач прочности, формируемых пакетом программ «ЛОГОС Прочность» [7]. Работа выполнена по заказу ФГУП “РФЯЦ-ВНИИЭФ”.

Статья построена следующим образом: приведена краткая постановка задачи решения СЛАУ, обоснован выбор MUMPS в качестве базового решателя, сформулированы основные идеи по оптимизации решателя с учетом блочной структуры матриц, показана важность выбора подходящей конфигурации запуска на кластере, приведены результаты вычислительных экспериментов и их анализ, в заключении обсуждены перспективы адаптации MUMPS для ускорителей GPU и Intel Xeon Phi.

1. Постановка задачи и метод решения

Пусть дана система линейных уравнений Ax = b, где A – разреженная симметричная положительно определенная матрица, b – плотный вектор, x – искомый вектор неизвестных. Прямые методы решения системы, как правило, основаны на применении разложения Холецкого к матрице A с последующим решением двух треугольных систем. При выполнении разложения Холецкого матрица обычно претерпевает заполнение, что на практике может привести к неудовлетворительным требованиям по памяти. Степень заполненности матрицы можно уменьшить с помощью переупорядочивания ее строк и столбцов. При решении системы с использованием разложения Холецкого можно выделить следующие этапы: переупорядочивание, символическое разложение (построение портрета фактора и выделение памяти для хранения ненулевых элементов), численное разложение (вычисление фактора) и решение треугольных систем. Численное разложение может выполняться одним из трех методов: ориентированным влево, ориентированным вправо, мультифронтальным. Подробное описание постановки задачи и мультифронтального метода приведено в работе [5].

2. Выбор MUMPS в качестве базового решателя

MUMPS (MUltifrontal Massively Parallel Solver) – программный комплекс для решения разреженных СЛАУ. Данный комплекс позволяет решать системы как с симметричными, так и с несимметричными матрицами. Причины выбора MUMPS в качестве базового решателя состоят в следующем:

1. MUMPS является одним из ведущих академических прямых решателей разреженных СЛАУ [11]. Он разрабатывается с 1996 г. в университетах Лиона, Тулузы, Бордо, регулярно обновляется, реализует обширную функциональность, имеет хорошую репутацию и большой спектр приложений [11].

2. MUMPS распространяется бесплатно в исходных кодах по лицензии Public Domain, что допускает его переработку сторонними группами и не накладывает существенных ограничений на использование.
3. MUMPS имеет высокопроизводительную MPI-реализацию для распределенных систем, дополнительные возможности по работе с большими матрицами (использование режима out-of-core), а также комбинированную версию для использования параллеллизма на общей и распределенной памяти. Все это позволяет решателю показывать хорошие результаты производительности в сравнении с другими академическими и коммерческими решателями.

3. Оптимизация MUMPS для работы с матрицами специального вида

Разреженные матрицы, возникающие в процессе решения трехмерных стационарных задач прочности методом конечных элементов, имеют нерегулярную разреженную структуру из плотных блоков размером 3×3 элемента. Такую матрицу размера N×N можно рассматривать как матрицу размера K×K, при K = N/3, где элементами являются плотные блоки размером 3×3. Основная идея выполненной оптимизации заключается в экономии времени при переупорядочивании матрицы с целью сокращения заполнения фактора. Экономия достигается за счет загрузки портрета матрицы, представленной в блочном формате, и его переупорядочивания средствами библиотек METIS/ParMETIS [3, 4] с последующим «расширением» полученного результата. При этом время работы стадии переупорядочивания сокращается (так, для матриц, состоящих из плотных блоков размера 3×3, число вершин в графе уменьшается в 3 раза). Кроме того, в ряде случаев может уменьшаться время работы на следующих стадиях алгоритма в связи с тем, что эвристический алгоритм поиска перестановки потенциально может найти лучшее решение в задаче меньшей размерности.

4. Выбор оптимальной конфигурации запуска

Выбор подходящей конфигурации запуска прикладного ПО, ориентированного на кластерные системы, часто весьма не прост. Так, например, при решении задачи нахождения собственных значений и векторов разреженной матрицы (в частности, при помощи известного пакета SLEPc) для каждой конкретной задачи требуется «угадать» достаточное количество используемых ядер, а также параметр, регулирующий размер создаваемых вспомогательных структур данных. Неудачное сочетание этих параметров может привести к аварийному завершению задачи в связи с нехваткой памяти, в связи с истечением лимита времени на одну задачу на кластере (либо неудовлетворительному времени счета при отсутствии лимита) и другим нежелательным последствиям. При решении разреженных СЛАУ прямыми методами неудачный выбор числа используемых процессоров обычно порождает аналогичные проблемы. Так, в некоторых случаях объемы требуемой оперативной памяти на один узел могут превышать объем доступной уста-новленной памяти, что вызывает аварийный останов или замедление работы на порядки в связи с использованием файла подкачки. В связи с указанными причинами задача выбора подходящей конфигурации запуска является достаточно актуальной.

В некоторых математических библиотеках и пакетах численного моделирования решению указанной проблемы уделяется значительное внимание. Так, например, в библиотеке ATLAS при сборке и установке выполняется калибровка параметров и настройка под конкретную вычислительную систему. Аналогичный подход применяется в пакете алгоритмов линейной алгебры MAGMA, ориентированном на работу с плотными матрицами на гетерогенных архитектурах (CPU + GPU, CPU + Xeon Phi).

Запуск специально подготовленных тестов для настройки под конкретное программно-аппаратное окружение не является единственным способом выбора параметров алгоритмов. Так, в некоторых библиотеках (Intel MKL, GOTO BLAS и др.) подобная настройка, судя по всему, выполняется при помощи определения параметров конкретной вычислительной системы (поддерживаемые наборы команд, размер кеш-памяти, длина кеш-линейки и др.) непосредственно при запуске. При этом можно применять разные подходы, а именно: параметризация программного кода, разработка разных реализаций для одной и той же функции с использованием разных наборов команд.

К сожалению, в обсуждаемых в данной работе алгоритмах решения разреженных СЛАУ выбор оптимального режима запуска затруднен. Конечно, настройка параметров основных математических ядер, в частности, умножения плотных матриц, выполняется аналогичными способами, но оценка размеров необходимой памяти, и тем более, оптимального соотношения числа процессов/потоков на одном узле является достаточно сложной задачей. Необходимо отметить, что MUMPS успешно справляется с оценкой необходимых объемов памяти без помощи пользователя. Тем не менее, выбор оптимального числа процессов MPI и потоков параллельного BLAS на вычислительном узле необходимо выполнять самостоятельно. В данной работе приводятся результаты экспериментов с распределенной версией MUMPS при варьировании числа процессов MPI/потоков OpenMP в BLAS, обсуждается возможность выбора подходящей конфигурации запуска.

5. Результаты экспериментов

Для анализа производительности программного комплекса были проведены эксперименты на кластере ФГУП “РФЯЦ ВНИИЭФ”, с использованием 1–36 узлов, 12 ядер на каждом узле. Характеристики тестовых матриц представлены в таблице ниже (таблица 1, столбцы 1–4). Эти СЛАУ получены из трехмерных стационарных задач прочности, выработанных пакетом программ «ЛОГОС Прочность» [7]. Все они являются разреженными, симметричными положительно определенными и имеют описанную ранее блочную структуру. Использовались ОС Linux, Intel MKL PARDISO, Intel C/C++ Compiler, Intel Fortran Compiler из пакета Intel

Parallel Studio XE 2011 и MUMPS 4.10.0. При сборке решателей использовался BLAS из Intel MKL, переупорядочиватель ParMETIS 4.0.

Результаты с использованием MKL PARDISO. Для получения базы для дальнейшего анализа результатов использовался решатель Intel MKL PARDISO из пакета Intel Parallel Studio XE 2011 в in-core-режиме для решения указанных задач. Intel MKL PARDISO не поддерживает распределенный режим работы, но демонстрирует отличные результат на одном узле кластера. Время работы решателя приведено в таблице 1 (столбец 4).

Таблица 1. Характеристики тестовых матриц. Результаты с использованием Intel MKL PARDISO

Название матрицы	Порядок	Число ненулевых элементов	Время решения с использованием 12 ядер, секунды
lopatka1	274 104	10 370 664	9,65
reshetka2	2 263 338	76 684 494	61,82
Trubka	2 428 323	70 338 539	62,08
49_750	2 615 169	96 358 289	Недостаточно памяти
p4_6	4 216 212	113 040 321	Недостаточно памяти

Intel MKL PARDISO позволяет решить две последние задачи в out-of-core-режиме с использованием жесткого диска в качестве «дополнительной памяти», однако сравнение результатов Intel MKL PARDISO в out-of-core-режиме и MUMPS в in-core-режиме (за счет большего объема памяти на нескольких узлах кластера) не имеет смысла.

Результаты оптимизации MUMPS для работы с матрицами специального вида. Рассмотрим результаты экспериментов для оригинальной версии MUMPS и версии MUMPS+, оптимизированной для работы с блочными матрицами (рисунок 1). На рисунке отображено время работы MUMPS и MUMPS+ при изменении числа используемых узлов от 2 до 36. По оси ординат отложено время в секундах, по оси абсцисс – число узлов кластера. Обе версии MUMPS запускались в режиме один MPI-процесс на один узел. На каждом узле создавалось 12 потоков (по числу вычислительных ядер).

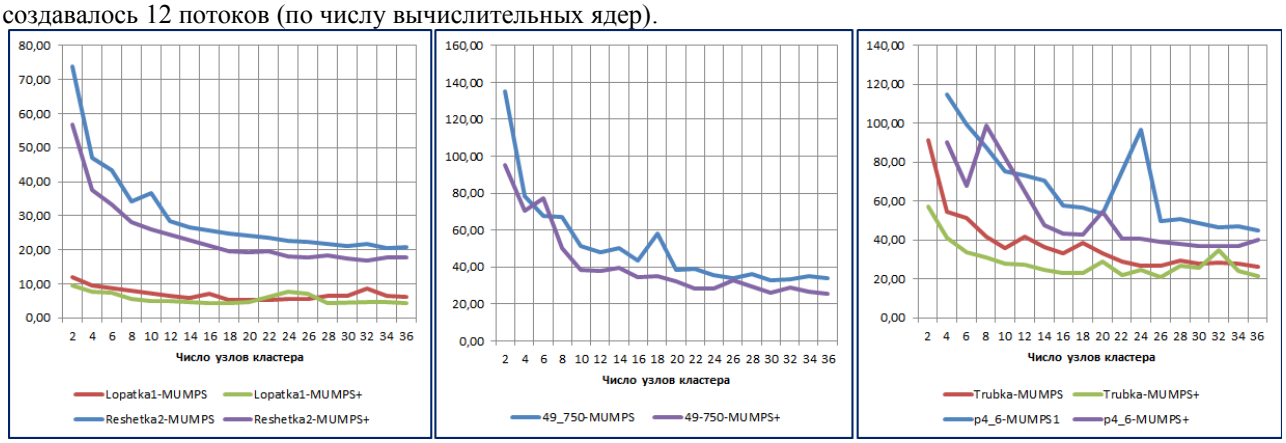

Рис. 1. Сравнение базовой и оптимизированной версий MUMPS

Из графиков видно, что при увеличении числа используемых узлов кластера время решения задачи ожидаемо уменьшается. Тем не менее, в некоторый момент наступает «насыщение», время стабилизируется и даже может начать расти. Этот эффект особенно заметен на матрице Lopatka1, что, впрочем, тоже ожидаемо: эта матрица имеет сравнительно небольшие размер и число ненулевых элементов. Начиная с некоторого момента, накладные расходы на поддержку параллельного режима работы становятся сопоставимы с выигрышем, получаемым от увеличения числа вычислительных ядер.

В таблице 2 представлено наименьшее время решения задачи с использованием MUMPS и MUMPS+ с указанием числа узлов, на котором оно достигается, а также время MUMPS+ на 2 узлах и для сравнения время Intel MKL PARDISO на 1 узле.

Таблица 2. Результаты с использованием Intel MKL PARDISO, MUMPS, MUMPS+

Название матрицы	Время решения, PARDISO, секунды	Время MUMPS+ на 2 узлах, секунды	Лучшее время, MUMPS, секунды	Лучшее время, MUMPS+, секунды
lopatka1	9,65	9,63	5,17 на 20 узлах	4,24 на 28 узлах
reshetka2	61,82	56,82	20,65 на 34 узлах	16,97 на 32 узлах
Trubka	62,08	57,08	26,04 на 36 узлах	20,72 на 26 узлах

49_750	Недостаточно памяти	95,34	32,58 на 30 узлах	25,66 на 36 узлах
p4_6	Недостаточно памяти	Недостаточно памяти	45,07 на 36 узлах	36,79 на 30 узлах

Результаты показывают, что при запуске MUMPS+ на 2 узлах удается улучшить результат Intel MKL PARDISO, запущенного на одном узле, что говорит о применимости оптимизированной версии MUMPS в сравнении с одним из лучших коммерческих решателей СЛАУ. Однако необходимо отметить, что описанный выше подход к оптимизации вычислений можно применить в любом прямом решателе, что должно привести к улучшению результатов при решении задач с блочной структурой.

Сравнение базовой и оптимизированной версий MUMPS показывает, что оптимизированная версия позволяет достигать в среднем на 20% лучшего времени решения задач с использованием меньшего числа вычислительных устройств.

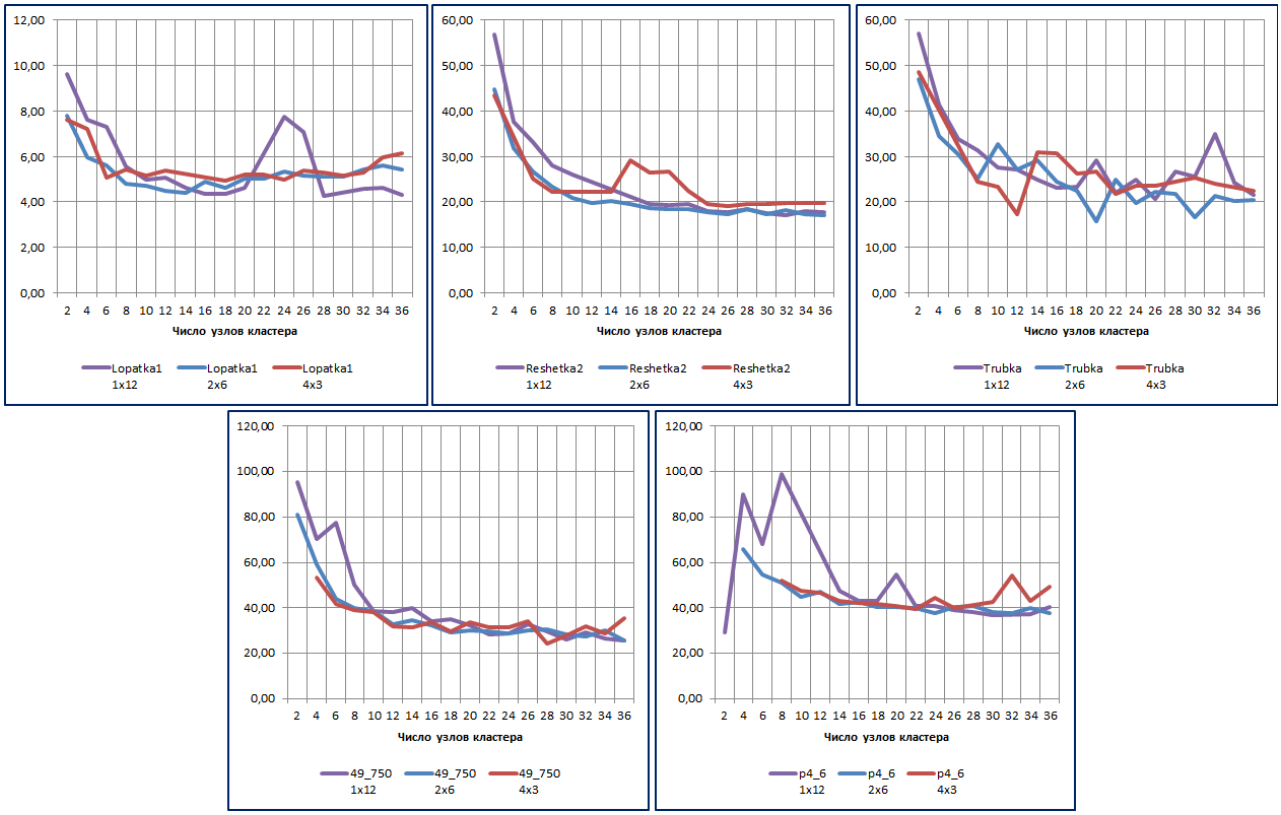

Рис. 2. Выбор оптимальной конфигурации запуска

Выбор оптимальной конфигурации запуска. Рассмотрим результаты работы по дальнейшему уменьшению времени счета при помощи выбора оптимального числа процессов/потоков на одном узле. Пусть Np – число процессов, Nt – число потоков BLAS, а Nc – число ядер. Будем выдерживать соотношение Nc = Np × Nt. Сравним 3 режима запуска оптимизированной версии MUMPS+: 1 × 12, 2 × 6, 4 × 3 (рисунок 2).

Наименьшее время показал вариант 2 × 6, при этом изменение режима запуска может значительно (до 2 раз) влиять на время работы, особенно при небольшом числе узлов. Это объясняется тем, что распараллеливание на потоках в MUMPS выполнено на уровне функций параллельного BLAS третьего уровня. Профилировка показывает, что ключевой операцией при этом является умножение плотных матриц, которое далеко не в каждом вызове эффективно распараллеливается на максимально возможное число потоков на узле. В таких случаях использование параллелизма на более высоком уровне может привести к сокращению общего времени счета. Необходимо отметить, что с ростом числа используемых узлов доля вычислений на узел сокращается, накладные расходы на передачи по сети увеличиваются, а время работы все меньше зависит от режима запуска.

Заключение

Обзор текущего положения дел в области решения СЛАУ с разреженной симметричной положительно определенной матрицей прямыми методами показывает наличие значительного числа программных комплексов, ориентированных как на системы с общей памятью, так и на кластерные системы, распространяемых как отдельно, так и в составе проблемно-ориентированных инженерных пакетов. Среди таких программных комплексов – один из лучших свободно распространяемых решателей с открытым кодом MUMPS, ориентированный на кластерные системы, и известный коммерческий решатель Intel MKL PARDISO, оптимизированный для многоядерных систем. В статье рассказывается об опыте оптимизации пакета MUMPS

для решения больших разреженных СЛАУ, возникающих при решении трехмерных стационарных задач прочности методом конечных элементов. Суть оптимизации заключается в учете блочной структуры задач на этапе переупорядочивания матрицы. Эксперименты показывают сокращение общего времени счета на ~20%. В работе изучается влияние режима запуска (число узлов кластера, соотношение числа MPI-процессов и потоков BLAS на одном узле) на время счета, демонстрируется существенное (до 2 раз при небольшом числе узлов) изменение времени счета в зависимости от кон-фигурации запуска. Оптимизированный решатель MUMPS подключен Заказчиком к комплексу библиотек параллельных решателей СЛАУ LParSol [12].

Направления дальнейшей деятельности могут быть связаны с переносом части вычислений на ускорители GPU и Intel Xeon Phi. Вопрос о возможности эффективной адаптации для ускорителей прямых решателей разреженных СЛАУ [13] изучается достаточно давно, но все еще является открытым. Один из наиболее простых способов адаптации в рассматриваемой задаче состоит в переносе функций BLAS на GPU или Xeon Phi. При этом возможно использование существующих высокопроизводительных библиотек (CUBLAS для GPU, MKL для Xeon Phi, MAGMA для гетерогенных систем). Указанный способ требует специального алгоритма выбора устройства для выполнения функций в зависимости от размера задачи. Большинство реализаций BLAS для ускорителей эффективны только для матриц достаточно большого размера. Очевидно, в большинстве случаев данный подход не приведет к ускорению вычислений в связи с малыми размерами матриц, обрабатываемых функциями BLAS. Исследование возможности создания эффективных реализаций рассмотренных алгоритмов для ускорителей является направлением дальнейшей работы.

ЛИТЕРАТУРА:

1. Amestoy P.R., Guermouche A., L'Excellent J.-Y., Pralet S. Hybrid scheduling for the parallel solution of linear systems // Parallel Computing Vol 32 (2), 2006. – P. 136-156.
2. Intel Math Kernel Library Reference Manual. – [http://software.intel.com/sites/products/documentation/hpc/mkl/mklman.pdf].
3. Karipis G. METIS. A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices. Version 5.0. // Technical report, University of Minnecota, Department of Computer Science and Engeneering. – 2011. – [http://glaros.dtc.umn.edu/gkhome/fetch/sw/metis/manual.pdf].
4. Karypis G., Kumar V. A fast and high quality multilevel scheme for partitioning irregular graphs // SIAM J. on Scientific Computing. – 1999. – Vol. 20, No. 1. – P. 359–392.
5. Liu J. The multifrontal method for sparse matrix solution: Theory and practice // SIAM Review, 34 (1992). – P. 82-109.
6. MUltifrontal Massively Parallel Solver (MUMPS 4.10.0) User's guide // Technical report ENSEEINT-IRIT. – 2011. – [http://mumps.enseeiht.fr/doc/userguide_4.10.0.pdf].
7. Речкин В.Н., Спиридонов В.Ф., Цибарев К.В., Дьянов Д.Ю., Наумов А.О., Косарим С.С., Филимонкин Е.А., Бартенев Ю.Г., Щаникова Е.Б., Ерзунов В.А., Рябов В.А., Вяткин Ю.А. Пакет программ ЛОГОС. Модуль решения квазистатических задач прочности и модального анализа // Труды XIII Международного семинара «Супервычисления и математическое моделирование» / Под ред. Р.М. Шагалиева. Саров, 2011.
8. Li X. Direct Solvers for Sparse Matrices. – [http://crd-legacy.lbl.gov/~xiaoye/SuperLU/SparseDirectSurvey.pdf]
9. Dongarra J. Freely Available Software for Linear Algebra. – [http://www.netlib.org/utk/people/JackDongarra/la-sw.html].
10. Intel MKL web page. – [http://software.intel.com/en-us/intel-mkl].
11. MUMPS web page. – [http://gral.ens-lyon.fr/MUMPS].
12. Бартенев Ю.Г., Бондаренко Ю.А., Ерзунов В.А. и др. Комплекс LParSol для решения СЛАУ // Сборник тезизов докладов XIII Международного семинара «Супервычисления и математическое моделирование». Саров, 2011, С.34-36.
13. Posey S. GPU progress in sparse matrix solvers for applications in computational mechanics // European seminar on computing, Pilzen, Chech Republic, June, 2012.