РАЗРАБОТКА ПРЯМОГО РЕШАТЕЛЯ ДЛЯ РАЗРЕЖЕННЫХ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ С СИММЕТРИЧНОЙ ПОЛОЖИТЕЛЬНО ОПРЕДЕЛЕННОЙ МАТРИЦЕЙ

Е.А. Козинов, И.Г. Лебедев, С.А. Лебедев, А.В. Линев, А.Ю. Малова, И.Б. Мееров, А.В. Сысоев, Т.А. Сысоева, С.С. Филиппенко, В.П. Гергель

Нижегородский госуниверситет им. Н.И. Лобачевского

Рассмотрен подход к решению СЛАУ с разреженной матрицей. Описана поэтапная схема решения СЛАУ с использованием метода Холецкого. Выполнена программная реализация представленной схемы. Приведены результаты экспериментов, дано их сравнение с результатами, полученными с помощью некоторых известных библиотек.

Введение

Одной из актуальных задач алгебры разреженных матриц является решение систем линейных алгебраических уравнений $Ax=b$ с разреженной матрицей A. Разработка алгоритмов решения таких задач и их высокопроизводительная программная реализация, ориентированная на современные вычислительные архитектуры, представляет большой практический интерес. Для решения разреженных СЛАУ применяются как прямые, так и итерационные методы. И те, и другие методы имеют свои недостатки, вытекающие из внутренней природы методов. Так, например, прямые методы требуют серьезные объемы дополнительной памяти для представления промежуточных матриц, а итерационные методы могут демонстрировать достаточно медленную сходимость к решению [2]. На практике выбор метода во многом зависит от решаемой задачи.

На сегодняшний день в мире разработано большое количество специализированного программного обеспечения для решения больших разреженных СЛАУ – так называемые «решатели» СЛАУ. В соответствии с используемыми методами эти решатели подразделяются на прямые и итерационные. Некоторые из них имеют высокопроизводительные реализации. В работе идет речь о разработке прямого решателя разреженных СЛАУ. Среди известных прямых решателей – MKL PARDISO, SuperLU, MUMPS, CHOLMOD и многие другие. Часть решателей ориентирована на последовательный режим работы, часть распараллелена для систем с общей памятью, часть «умеет» работать в системах с распределенной памятью. Часть решателей ориентирована только на симметричные положительно определенные матрицы (что существенно упрощает дело, решая некоторые численные проблемы), часть – на матрицы общего вида. Некоторые из решателей поддерживают так называемый out-of-core режим работы, используя жесткий диск в качестве «продолжения» оперативной памяти. Хороший, постоянно обновляемый обзор прямых решателей от авторов SuperLU можно скачать по следующей ссылке: http://crd.lbl.gov/~xiaoye/SuperLU/SparseDirectSurvey.pdf.

Задачей коллектива авторов является разработка собственной реализации высокопроизводительного прямого решателя разреженных СЛАУ с симметричной положительно определенной матрицей A. Мотивация для создания своего инструмента состоит в потенциальной возможности создания конкурентноспособного инструмента, а также перспективах использования в учебном процессе.

В работе приведены текущие результаты, дано их сравнение с результатами некоторых известных библиотек, а также определены пути дальнейшего развития.

Постановка задачи

Пусть дана система линейных уравнений:

$$Ax = b \quad (1)$$

Здесь A – разреженная симметричная положительно определенная матрица, b – плотный вектор, x – вектор неизвестных. Необходимо найти решение системы x.

Метод решения

Прямые методы решения задачи (1) как правило основаны на применении разложения Холецкого к матрице A в виде

$$A = U^T U, \quad (2)$$

где U – верхнетреугольная матрица. В этом случае решение системы сводится к последовательному решению двух треугольных систем:

$$U^T y = b, \quad (3)$$

$$Ux = y. \quad (4)$$

Особенностью процедуры разложения Холецкого для разреженной матрицы является то, что матрица обычно претерпевает заполнение, что на практике может привести к неудовлетворительным требованиям по памяти. Степень заполненности матрицы можно уменьшить переупорядочиванием ее строк и столбцов. Это соответствует нахождению матрицы перестановки P и переходу к эквивалентной системе (6):

$$\bar{A} = PAP^T, \quad (5)$$

$$\bar{A}(Px) = PB. \quad (6)$$

Таким образом, при решении разреженной системы с использованием метода Холецкого можно выделить следующие этапы:

1) переупорядочивание – вычисление матрицы перестановки P и переход к системе (6);
2) символическое разложение – построение портрета матрицы U, выделение памяти для хранения ненулевых элементов;
3) численное разложение – вычисление значений матрицы U и размещение их в выделенной памяти;
4) обратный ход – решение треугольных систем (3), (4).

Программная реализация

На основе данной схемы выполнена последовательная программная реализация решателя СЛАУ, а также подготовлен прототип параллельной версии для систем с общей памятью. Программная реализация выполнена на языке C. Для хранения матрицы разреженных матриц выбран формат CSR. Вычисления проводились с двойной точностью. Рассмотрим подробнее реализацию каждого этапа метода.

Известны разные подходы для оптимизации заполнения фактора, в частности, метод минимальной степени, метод вложенных сечений и другие. Метод вложенных сечений (nested dissection) был предложен А. Джорджем в 1973 г. и основан на разбиении графа матрицы при помощи разделителей. По сравнению с методом минимальной степени, он позволяет достичь большей степени параллеллизма на стадиях факторизации. Реализация алгоритма вложенных сечений была выполнена в соответствии с описанием алгоритма в книге [2].

После получения перестановки дополнительно применяется пост-переупорядочивание в соответствии с алгоритмом, приведенным в [1]. Этот прием повышает структурированность портрета исходной матрицы и фактора, не ухудшая его заполненности. Применение пост-переупорядочивания позволяет ускорить численную фазу факторизации.

В рамках символической фазы разложения можно выделить два этапа. Сначала оценивается общее число ненулевых элементов в факторе и выделяется память для массива столбцовых индексов. Оценка выполняется в соответствии с алгоритмом,

предложенным J. Gilbert, E. Ng, и B. Peyton в [1]. Затем определяется портрет фактора матрицы U на основе процедуры, описанной в [3]. Данный алгоритм получает портрет каждой строки с неупорядоченными по возрастанию номерами столбцов. Для приведения к упорядоченному формату используется двойное транспонирование.

Во время численной фазы выполняется нахождение значений элементов верхнего треугольника. Это самая трудоемкая часть факторизации, т.к. для вычисления i-й строки фактора нужно знать значения в строках, содержащих ненулевой элемент в i-м столбце. Численная часть, а также обратный ход были реализованы в соответствии с описанием алгоритмов в [3].

Для нахождения числа ненулевых элементов в факторе, а затем и для выполнения параллельных вычислений, используется специальная структура – дерево исключения. Деревом исключения $T(A)$, соответствующим матрице A, называется дерево, множество вершин которого совпадает с множеством вершин графа матрицы (т.е. множеством строк), а множество ребер $E(T)$ задается соотношением:

$$e_{i,j} \in E(T) \leftrightarrow i = \min_k \{u_{jk} \neq 0 \;\&\; k > j\}, \quad (7)$$

где u_{jk} – элемент фактора матрицы.

Дерево исключения отражает зависимость между строками матрицы: если вершины не соединены ребром, то соответствующие им строки фактора могут вычисляться параллельно. В рассматриваемой реализации выполняется приближенное построение дерева исключения по мере нахождения перестановки.

Результаты экспериментов

Для анализа производительности программного комплекса был проведен ряд экспериментов на матрицах из коллекции [4] университета Флориды. В табл. 1 приведены характеристики выбранных матриц. Все они являются симметричными положительно определенными.

Таблица 1. Характеристики тестовых матриц

Название матрицы	Порядок	Число ненулевых элементов	Заполненность, %
pwtk	217 918	5 926 171	0,0125
msdoor	415 863	10 328 399	0,0060
parabolic_fem	525 825	2 100 225	0,0008
tmt_sym	726 713	2 903 837	0,0005
G3_circuit	1 585 478	4 623 152	0,0002
ecology2	999 999	2 997 995	0,0003
audikw_1	943 695	39 297 771	0,0044

Параметры тестовой инфраструктуры приведены в табл. 2.

Таблица 2. Параметры тестового окружения

Процессор	2 четырехъядерных процессора Intel® Xeon E5520 (2.27 GHz)
Память	16 Gb
Операционная система	Linux
Среда разработки	Microsoft Visual Studio 2008
Компилятор	Intel® Parallel Studio XE 2011

В табл. 3 приведены результаты запусков последовательной версии решателя на тестовых матрицах, выделено время работы каждого этапа.

Таблица 3. Время работы последовательной реализации (секунды)

Матрица	Число ненулевых эле-	Переупорядочи-	Символическая часть	Численная часть	Обратный ход	Общее время

	ментов в факторе	вание				
pwtk	61 678 703	0,75	1,96	47,33	0,3	50,58
msdoor	180 142 589	2,39	7,44	386,93	0,88	398,13
parabolic_fem	26 872 825	1,06	0,72	7,43	0,14	9,49
tmt_sym	45 181 460	4,58	1,29	19,93	0,24	26,19
G3_circuit	501 907 094	174,42	22,95	1 393,54	2,2	1 593,4
ecology2	33 203 232	53,58	0,93	10,19	0,17	65,02
audikw_1	Недостаточно памяти для хранения фактора					

Реализованный решатель отработал на всех выбранных матрицах, кроме audikw_1. На ней программа завершила работу, т.к. не хватило ресурсов системы для хранения фактора.

Также был проведен ряд экспериментов на тех же тестовых матрицах с использованием следующих библиотек:

- MKL PARDISO: Intel® Math Kernel Library (в составе Intel® Parallel Studio XE 2011);
- SuperLU Version 4.1.

Результаты экспериментов приведены в табл. 4.

Для пакета SuperLU приведено время работы при одном из встроенных алгоритмов перестановки, дающим меньшее время работы. Для матриц pwtk, parabolic_fem, tmt_sym, ecology2 это приближенный столбцовый метод минимальной степени (COLAMD), для матрицы msdoor – множественный метод минимальной степени (MMD). Отметим, что на матрице G3_circuit программа завершила работу из-за внутренней ошибки работы с памятью; на матрице audikw_1 – в связи с нехваткой ресурсов для хранения фактора.

Таблица 4. Сравнение работы решателей на тестовых матрицах

Матрица	Решатель авторов		SuperLU		SuperLU, 8 потоков		MKL		MKL, 8 потоков	
	Число элементов в факторе	t, сек	Число элементов в факторе	t, сек	Число элементов в факторе	t, сек	Число элементов в факторе	t, сек	Число элементов в факторе	t, сек
pwtk	61 678 703	50,58	108904188	79,81	109716912	13,59	52190144	5,45	51050753	1,45
msdoor	180 142 589	398,13	111245898	80,47	111395559	18,68	57478740	6,06	57096124	1,90
parabolic_fem	26 872 825	9,49	52361272	37,59	54270042	7,38	28125024	6,49	27941564	2,52
tmt_sym	45 181 460	26,19	88165169	76,89	88177959	14,77	33075548	8,66	32816445	3,48
ecology2	33 203 232	65,02	68677613	38	68688337	8,01	38516392	10,44	38876742	4,48
G3_circuit	501 907 094	1593,4	Ошибка работы с памятью				104692916	25,47	102828595	8,67
audikw_1	Недостаточно памяти для хранения фактора		Недостаточно памяти для хранения фактора				1270292396	793,03	1270735946	146,07

Как видно из табл. 4, последовательная версия решателя показывает результаты, сравнимые с SuperLU, и значительно отстающие от MKL (более чем на порядок). Т.к. наиболее трудоемким этапом вычислений является численная факторизация, время работы программ в значительной мере определяется полученным числом ненулевых элементов в факторе, которое зависит от качества работы метода перестановки.

Реализация авторов работает быстрее реализации из библиотеки SuperLU на матрицах parabolic_fem, tmt_sym, pwtk. На всех тестовых примерах размер фактора получен меньше, чем у SuperLU (в среднем в 1,7 раз), что позволило сократить время работы численной части. Наибольший выигрыш по времени был получен на матрице parabolic_fem: фактор меньше в 1,9 раз, время работы меньше в 4 раза, наибольший проигрыш – на матрице msdoor – в 4,9 раз (фактор больше на 40%).

Отставание последовательной версии решателя по отношению к MKL в среднем составляет 16,5 раз на всех матрицах, за исключением msdoor: от 1,5 на parabolic_fem до 62,6 на G3_circuit, на msdoor – 65,7 раз. Размеры факторов матриц, полученных в результате переупорядочивания, отличаются в среднем в 2 раза (до 4,8 раз на G3_circuit), причем на матрицах parabolic_fem и ecology2 перестановка MKL дает больше ненулевых элементов.

Заключение

На данный момент авторами реализована базовая последовательная версия решателя для систем с общей памятью. Как видно из результатов экспериментов, полученная реализация дает результаты, сравнимые с результатами SuperLU. На матрицах размером до миллиона строк отставание от MKL не превышает одного порядка, однако на больших размерах отставание нарастает.

Целью дальнейшего исследования является модификация последовательной версии для увеличения ее производительности, а также разработка полнофункциональной параллельной версии решателя. Основное направление работ – реализация более эффективных алгоритмов для каждой фазы решателя, в первую очередь – численной фазы и переупорядочивателя, а также достижение приемлемой масштабируемости для параллельной версии решателя для систем с общей памятью.

Работа выполнена в лаборатории «Информационные технологии» ВМК ННГУ.

Литература

1. Gilbert J.R., Ng E.G., Peyton B.W. An efficient algorithm to compute row and column counts for sparse Cholesky factorization // SIAM J. Matrix Anal. Appl. 1994. Vol. 15, №4. P. 1075-1091. October.
2. Джордж А., Лю Дж. Численное решение больших разреженных систем уравнений. – М.: Мир, 1984.
3. Писсанецки С. Технология разреженных матриц. – М.: Мир, 1988.
4. The University of Florida Sparse Matrix Collection – [www.cise.ufl.edu/research/sparse/matrices/].