Параллельный алгоритм построения QR-разложения разреженных матриц для систем с общей памятью

А.Е. Новак, С.А. Лебедев

Нижегородский государственный университет им. Н. И. Лобачевского

Построение QR-разложения больших разреженных матриц является одним из вычислительно трудоемких этапов при решении многих прикладных задач. В работе предложено использовать двухуровневый параллельный алгоритм для выполнения численной фазы мультифронтального QR-разложения. Алгоритм основан на использовании механизма логических задач OpenMP. Программная реализация алгоритма выполнена на языке программирования C++ с использованием высокопроизводительных функций BLAS.

Ключевые слова: разреженные матрицы, QR-разложение, мультифронтальный метод.

1. Введение

Необходимость решения задач линейной алгебры возникает в большом количестве научно-технических расчётов. Так, например, численное решение дифференциальных уравнений в частных производных, моделирующих различные физические процессы, сводится к решению систем линейных алгебраических уравнений вида $Ax = b$. Важной особенностью является разреженность и большие размеры матрицы A, которые могут достигать десятков миллионов строк и столбцов. Поэтому большой интерес представляют эффективные методы хранения и обработки таких матриц.

QR-разложение разреженных матриц может использоваться для прямого решения систем линейных алгебраических уравнений (СЛАУ), а также для решения задачи наименьших квадратов.

Решение системы алгебраических уравнений (СЛАУ) сводится к решению двух других, более простых систем. Важным достоинством этого подхода является то, что QR-разложение существует для любой матрицы A. Таким образом, круг задач, допускающих решение данным методом, достаточно широк. К сожалению, данная процедура является вычислительно трудоемкой и затратной по памяти. Это обуславливает интерес к разработке подходов к ее эффективной реализации на современных вычислительных системах.

На сегодняшний день применяется несколько методов для вычисления QR-разложения разреженных матриц [3]. Для ряда методов существуют достаточно эффективные программные реализации [1, 4]. В данной работе приводится обзор существующих методов, а также предлагается параллельная программная реализация для систем с общей памятью, основанная на механизме логических задач OpenMP.

2. Постановка задачи

Пусть дана система линейных уравнений $Ax = b$, где A – разреженная матрица размера $m \times n$, b – плотный вектор, x – вектор неизвестных. Для нахождения решения системы x к матрице A применяется QR разложение в виде $A = QR$, где Q – ортогональная матрица порядка $m \times m$ ($Q^{-1} = Q^T$), R – верхняя треугольная матрица порядка $n \times n$. Такое разложение существует для любой матрицы A. Кроме того, в случае введения дополнительных следующих ограничений (A – невырожденная матрица, элементы на диагонали R – положительные вещественные числа) разложение существует и единственно.

Если построено QR-разложение матрицы A, СЛАУ $Ax = b$ можно записать в виде: $QRx = b$ и свести решение исходной системы к двум другим:

$$\begin{cases} Qy = b, \\ Rx = y. \end{cases}$$

Для решения системы $Rx = y$ необходимо применить достаточно простой алгоритм, так как матрица R является верхней треугольной. Решение y системы $Qy = b$ может быть найдено в виде $Q^T b$ в силу ортогональности матрицы Q.

3. Последовательный алгоритм

Для решения поставленной задачи могут быть применены различные методы. Их можно разделить на несколько категорий. Во-первых, по типу преобразования, применяемого для исключения элементов ниже главной диагонали в ведущем столбце, методы делятся на использующие отражение Хаусхолдера и использующие вращение Гивенса. Во-вторых, в случае разреженной матрицы важным фактором является выбор момента применения преобразования к еще необработанной части исходной матрицы. В этой категории методы делятся на ориентированный влево, ориентированный вправо и мультифронтальный. Подробный анализ каждого из этих методов можно найти в работе [8]. Основным преимуществом мультифронтального метода является то, что он в большей степени, чем остальные, подходит для современных вычислительных архитектур. Мультифронтальный метод в процессе своей работы может эффективно использовать иерархическую систему памяти, а также векторные инструкции процессора, что значительно увеличивает его производительность по сравнению с остальными методами. Кроме того, можно отметить наличие большого потенциала использования параллельных вычислений в мультифронтальном методе. К недостаткам метода можно отнести большое потребление памяти, характерное для всех прямых методов разреженной алгебры.

Авторами мультифронтального метода построения QR-разложения являются Дафф и Рейд [2]. Изначально данный метод был разработан для разложения Холецкого и LU-разложения и затем успешно модифицирован для получения QR-разложения. Основная идея метода заключается в сведении исходной задачи к обработке серии небольших матриц, называемых фронтальными. Переход к операциям над плотными подматрицами позволяет эффективнее использовать память. Зависимость между подзадачами определяется по специальному графу (дереву исключения), позволяющему выделить независимые операции, которые могут быть выполнены параллельно. Эффективность параллельной реализации может быть оценена на начальном этапе. Мультифронтальный алгоритм состоит из двух фаз: символьная и численная.

Первая фаза называется символьной, так как использует только портрет матрицы A. На данном этапе происходит построение дерева исключения и определение портретов матриц Q и R. Иными словами, символьная часть осуществляет подготовку для последующих численных расчётов. Символьная фаза закладывает фундамент для всех вычислений данного метода. Правильно построенное дерево исключения позволяет повысить эффективность параллельной работы программы. Знание портретов матриц значительно упрощает дальнейшие вычисления.

Особенность, на которую стоит обратить внимание, – это характер заполненности матрицы Q. Зачастую она оказывается плотной, а число ненулевых элементов в матрице R становится достаточно большим. Этот факт значительно увеличивает затраты времени и памяти, необходимые для нахождения QR-разложения. Одним из способов решения данной проблемы является переупорядочение исходной матрицы. Под переупорядочением будем понимать процедуру, переставляющую строки и столбцы матрицы с целью уменьшения заполнения фактора [5]. Для нахождения такой перестановки строк и столбцов можно использовать различные программные решения или реализовать собственные алгоритмы. Среди программных продуктов наиболее популярными являются решения от авторов Metis и SuiteSparse. Среди алгоритмов можно выделить следующие: COLAMD, CHOLMOD и другие. Однако важно понимать, что вычисление такой перестановки требует дополнительных затрат времени и памяти. Программному пакету Metis необходимо знать матрицу AA^T что требует $O(N^3)$ вычислений и может занимать больше времени, чем само вычисление QR-разложения. В свою очередь, метод COLAMD, предложенный Дэвисом в своей работе [6], может применять перестановку к неквадратной матрице и не требует столь больших вычислительных затрат.

Вторая фаза является основной частью мультифронтального метода. В ней выполняются вычисления, связанные с обработкой фронтальных матриц. Как результат, матрицы Q и R за-

полняются числовыми значениями. Будем называть блоком дополнения все строки фронтальной матрицы, кроме начальной, с отсеченными первыми элементами.

Рассмотрим алгоритм численной фазы мультифронтального QR-разложения матрицы A.

Алгоритм 1. Высокоуровневое описание мультифронтального метода QR-разложения

Цикл: Для каждого узла k **дерева исключения** в **топологическом порядке**
  Построить фронтальную матрицу F_k следующим образом:
  Если k – лист
    F_kсоставляется из всех тех строк исходной матрицы, у которых в столбце с номером k содержится ненулевой элемент
  Иначе
    F_k составляется из всех тех строк исходной матрицы, у которых в столбце с номером k содержится ненулевой элемент, с присоединением в конец прямоугольных «блоков дополнения» (contribution block), полученных при обработке узлов-потомков дерева исключения
  Исключить во фронтальной матрице в столбце с номером k элементы под диагональю при помощи отражения Хаусхолдера
  Первая строка во фронтальной матрице есть k-ая строка матрицы R
  Оставшаяся после удаления во фронтальной матрице первой строки и первого столбца подматрица C_k есть блок дополнения, который отправляется в конец фронтальной матрицы узла-предка дерева исключения.
Конец цикла

Время работы численной фазы существенно зависит от размеров блоков дополнения. Одним из способов оптимизации является применение триангуляризации, которая представляет собой приведение фронтальных матриц к верхней треугольной форме. Триангуляризация достигается посредством применения отражений Хаусхолдера к первым k столбцам фронтальной матрицы (при $k = \min(m, n)$, где m, n — число строк и столбцов фронтальной матрицы). Заметим, что в случае $m < n$ фронтальная матрица будет приведена не к верхней треугольной форме, а к верхней трапециевидной. Данная операция позволяет уменьшить размер блока дополнения некоторых столбцов, что существенно уменьшает размеры фронтальных матриц.

4. Параллельный алгоритм

Для распараллеливания численной фазы мультифронтального метода может применяться распараллеливание по задачам, а также распараллеливание плотных операций BLAS с фронтальными матрицами в процессе работы метода. В качестве основного способа распараллеливания в данной работе используется параллельная обработка узлов дерева исключения. Для этого основной вычислительный цикл мультифронтального метода (Алгоритм 1) может быть выполнен параллельным образом с использованием логических задач OpenMP. Зависимости между задачами описываются тем фактом, что родительские узлы в дереве исключения не могут быть выполнены раньше своих дочерних узлов. Данная параллельная схема изначально была предложена для мультифронтального метода разложения Холецкого [10] и в данной работе адаптирована для QR-разложения.

Алгоритм 2. Параллельный мультифронтальный метода QR-разложения

Процедура *шаг_мультифронтального метода* (номер узла **k**)
  Построить фронтальную матрицу F_k следующим образом:
  Если k – лист
    F_kсоставляется из всех тех строк исходной матрицы, у которых в столбце с номером k содержится ненулевой элемент

5 **Иначе**

6 F_k составляется из всех тех строк исходной матрицы, у которых в столбце с номером k содержится ненулевой элемент, с присоединением в конец прямоугольных «блоков дополнения» (contribution block), полученных при обработке узлов-потомков дерева исключения

7 Исключить во фронтальной матрице в столбце с номером k элементы под диагональю при помощи отражения Хаусхолдера

8 Первая строка во фронтальной матрице есть k-ая строка матрицы R

9 Оставшаяся после удаления во фронтальной матрице первой строки и первого столбца подматрица C_k есть блок дополнения, который отправляется в конец фронтальной матрицы узла-предка дерева исключения

10 **Конец процедуры**

11

12 **Процедура** обход_дерева_исключения(номер узла k)

13 **Цикл:** Для всех детей **c** узла k

14 #pragma omp task

15 обход_дерева_исключения(**c**)

16 **Конец цикла**

17 #pragma omp taskwait

18 *шаг_мультифронтального метода(k)*

19 **Конец процедуры**

В параллельном алгоритме мультифронтального метода QR-разложения (Алгоритм 2) выполняется обход узлов дерева исключения, при этом для каждого узла порождаются отдельные логические задачи для всех детей узла. Выполнение шага мультифронтального метода для узла приостанавливается до тех пор, пока все порожденные задачи не будут выполнены. Нужно отметить что вычислительная трудоемкость для обработки каждого узла дерева исключения различна. Поэтому указанный подход позволяет динамически распределять нагрузки между вычислительными ядрами в процессе своей работы.

5. Результаты вычислительных экспериментов

5.1 Методика проведения экспериментов

Для проведения вычислительных экспериментов использовался узел кластера ННГУ «Лобачевский» (Таблица 1). Программный код был скомпилирован с использованием пакета Intel Parallel Studio XE 2017, для выполнения операций с плотными матрицами, возникающими в процессе работы метода, использовались библиотеки MKL BLAS и MKL LAPACK.

Таблица 1. Характеристики вычислительного узла

Процессор	2 x Intel Xeon CPU E5-2660 2.20 GHz, 8 ядер
Оперативная память	64 GB RAM
Операционная система	CentOS Linux 7

Таблица 2 содержит характеристики тестовых матриц из набора SuiteSparse Matrix Collection [7], на которых проводились запуски. Перестановка для минимизации заполнения в фактор-матрице R была получена с помощью библиотеки Metis [9].

Таблица 2. Характеристики тестовых матриц

Матрица	**Описание**	**Размер (N)**	**Число ненулевых элементов в исходной матрице A**	**Число ненулевых элементов в фактор-матрице R**

parabolic_fem	Computational Fluid Dynamics Problem	525 825	3 674 625	84 522 298
engine	3D solid model, thermal stress analysis of engine head	143 571	4 706 073	81 086 064
linverse	Statistical/Mathematical Problem	11 999	95 977	190 627
Ill_Stokes	Ill-conditioned matrix from a Stokes problem	20 896	191 368	9 895 686
tmt_sym	Symmetric electromagnetics problem	726 713	5 080 961	118 077 178
tmt_unsym	Unsymmetric electromagnetics problem	917 825	4 584 801	105 539 699
ecology2	Circuitscape: circuit theory applied to animal/gene flow	999 999	4 995 991	125 653 200
offshore	Transient electric field diffusion	259 789	4 242 673	389 610 878

5.2 Анализ производительности

Для анализа производительности численной фазы QR-разложения были проведены запуски разработанного решателя на каждой из тестовых матриц. В каждом запуске число параллельных потоков изменялось от 1 до 16. Результаты вычислительных экспериментов представлены в таблице 3, время работы дано в секундах. Ускорение численной фазы приведено на рисунке 1.

Таблица 3. Время работы численной фазы мультифронтального QR разложения (сек.)

Матрица	**Количество потоков**				
	1	2	4	8	16
parabolic_fem	18,84	10,81	6,86	4,90	4,10
engine	32,65	17,16	12,83	6,83	6,11
linverse	0,06	0,04	0,03	0,03	0,03
Ill_Stokes	5,36	3,69	2,70	2,39	2,23
tmt_sym	30,48	16,47	10,59	7,74	7,08
tmt_unsym	26,63	15,43	9,67	7,92	7,17
ecology2	34,63	20,74	13,43	9,81	9,77
offshore	1589,63	975,09	594,85	501,19	421,88

Из результатов вычислительных экспериментов можно сделать следующие выводы. Для двух потоков среднее значение ускорения составляет 1,67. При этом, если исключить из рассмотрения две самые маленькие матрицы (linverse и Ill_Stokes), среднее значение ускорения будет равно 1,75. Указанные матрицы содержат слишком мало ненулевых элементов, что приводит к малым размерам одной логической задачи и, как следствие, малой эффективности распараллеливания. В дальнейшем, при описании результатов, исключим эти матрицы из рассмотрения. При увеличении числа потоков эффективность распараллеливания начинает снижаться, ускорение составляет в среднем 3,77 и 4,21 для 8 и 16 потоков соответственно. Это объясняется тем, что при увеличении числа потоков и распараллеливании с использованием логических задач при достижении определенного уровня в дереве исключения количество независимых задач уменьшается и потоки находятся в состоянии ожидания. Кроме того, при приближении к корню дерева исключения, как правило, размер задач становится больше. Для решения данной проблемы может быть применена двухуровневая схема [11], в которой, начиная с некоторого слоя, происходит переключение с параллелизма на основе задач на параллелизм с использованием параллельных процедур библиотек BLAS и LAPACK. Внедрение подобной схемы является одним из направлений дальнейшей работы.

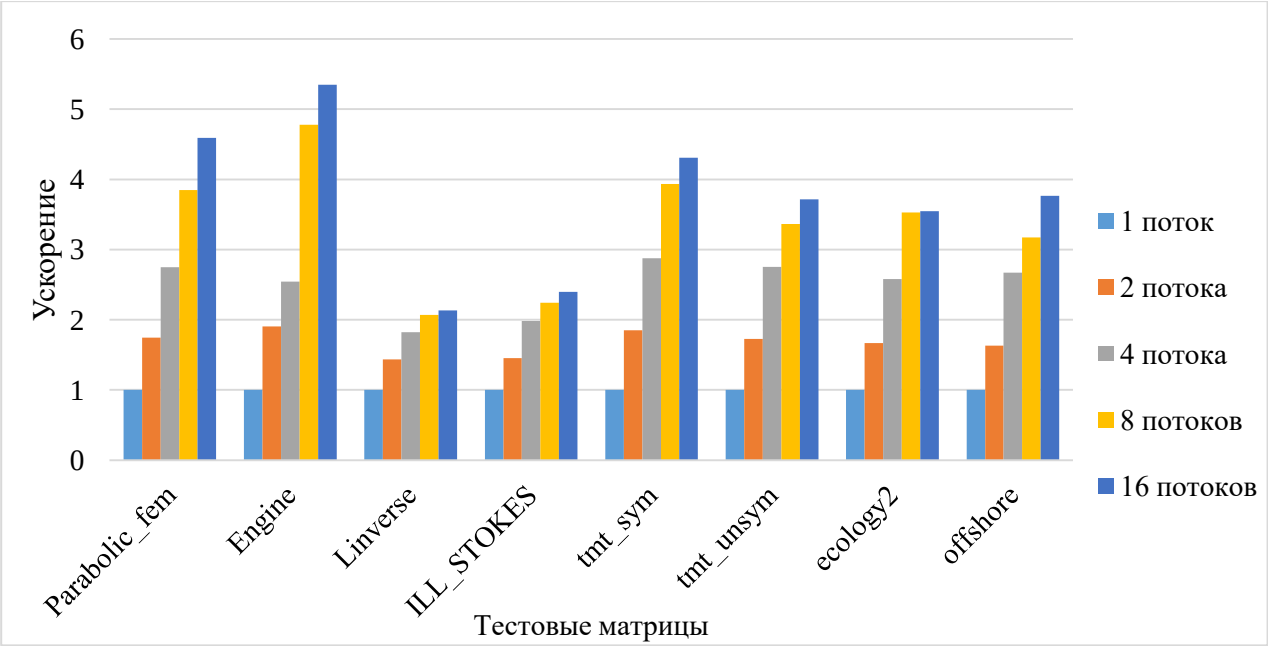

Рис. 1. Ускорение численной фазы мультифронтального QR разложения

Заключение

В работе представлено описание параллельного алгоритма выполнения численной фазы QR-разложения разреженных матриц. Последовательный алгоритм следует схеме вычислений мультфронтального метода, широко используемого для факторизации разреженных матриц. Параллельный алгоритм формирует последовательность логических подзадач в схеме мультифронтального метода. В качестве таких подзадач рассматриваются вычисления, соответствующие вычислению строки фактора R. Параллельная обработка подзадач выполняется при помощи механизмов OpenMP 4.0. Первые результаты экспериментов на широко распространенной коллекции матриц Suite Sparse (ранее – коллекция матриц Университета Флориды) показывают, что алгоритм приемлемо масштабируется до 4–8 ядер. При переходе к 16 ядрам время работы сокращается, но эффективность масштабируемости падает в среднем до 25%. Далее планируется продолжить работу в направлении повышения эффективности используемой схемы распараллеливания при помощи адаптивного выбора момента переключения между параллелизмом по задачам и использованием параллельного BLAS. Дальнейшее сокращение числа ненулевых элементов в факторе R путем улучшения используемых перестановок столбцов исходной матрицы является одним из перспективных направлений сокращения затрат времени и памяти при развитии рассмотренного алгоритма.

Литература

1. Agullo E. et al. Exploiting a Parametrized Task Graph model for the parallelization of a sparse direct multifrontal solver //European Conference on Parallel Processing. – Springer, Cham, 2016. P. 175-186..
2. Amestoy P. R., Duff I. S., Puglisi C. Multifrontal QR factorization in a multiprocessor environment //Numerical linear algebra with applications. 1996. V. 3. No. 4. P. 275-300.
3. Björck Å. Numerical Methods in Matrix Computations, 2015 - Springer. P 246-291
4. Davis T. A. Algorithm 915, SuiteSparseQR: Multifrontal multithreaded rank-revealing sparse QR factorization //ACM Transactions on Mathematical Software (TOMS). 2011. Vol. 38. No. 1. – P.
5. Davis T.A., Gilbert J.R., Larimore S., Ng E. A column approximate minimum degree ordering algorithm // Transactions on Mathematical Software. – 2004. – Vol. 30, No. 3. – P. 353–376.

6. Davis T. A. et al. Algorithm 836: COLAMD, a column approximate minimum degree ordering algorithm //ACM Transactions on Mathematical Software (TOMS). 2004. V. 30. No. 3. P. 377-380.
7. Davis T. A., Hu Y. The University of Florida sparse matrix collection //ACM Transactions on Mathematical Software (TOMS). 2011. V. 38. No. 1. P. 1.
8. Davis T. A., Rajamanickam S., Sid-Lakhdar W. M. A survey of direct methods for sparse linear systems //Acta Numerica. 2016. V. 25. P. 383-566.
9. Karypis G. and Kumar V. ParMetis: Parallel graph partitioning and sparse matrix ordering library. Tech. Rep. TR 97-060, University Of Minnesota, Department Of Computer Science. 1997
10. Lebedev S. et al. Dynamic parallelization strategies for multifrontal sparse cholesky factorization //International Conference on Parallel Computing Technologies. – Springer, Cham, 2015. – P. 68-79.
11. Лебедев С. А. и др. Двухуровневый параллельный алгоритм выполнения численной фазы разложения Холецкого для разреженных матриц //Вестник Уфимского государственного авиационного технического университета. 2015. Т. 19. №. 3 (69).