УДК 519.61 DOI: 10.14529/cmse160203

ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА РАЗРЕЖЕННОГО QR РАЗЛОЖЕНИЯ ДЛЯ ПРЯМОУГОЛЬНЫХ ВЕРХНИХ КВАЗИТРЕУГОЛЬНЫХ МАТРИЦ СО СТРУКТУРОЙ РАЗРЕЖЕННОСТИ ТИПА ВЛОЖЕННЫХ СЕЧЕНИЙ[1]

С.А. Харченко, А.А. Ющенко

В работе рассматривается параллельная MPI+threads+SIMD реализация алгоритма вычисления разреженного QR разложения специальным образом упорядоченной прямоугольной матрицы на основе разреженных блочных преобразований Хаусхолдера. В алгоритме производится предварительное независимое параллельное вычисление QR разложений для наборов строк матрицы. Затем в соответствии с деревом вычислений производится вычисление QR разложения матриц, составленных из R факторов строчных разложений. Приводятся результаты экспериментов, подтверждающие эффективность предложенной параллельной реализации для тестовых задач. Алгоритм также может быть эффективно реализован на гетерогенных кластерных архитектурах с ускорителями типа GPGPU.

Ключевые слова: разреженная, прямоугольная матрица, верхняя квазитреугольная матрица, QR разложение, вложенные сечения, преобразования Хаусхолдера, MPI, многопоточность, SIMD.

ОБРАЗЕЦ ЦИТИРОВАНИЯ

Харченко С.А., Ющенко А.А. Параллельная реализация алгоритма разреженного QR разложения для прямоугольных верхних квазитреугольных матриц со структурой разреженности типа вложенных сечений // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. 2016. Т. 5, № 2. С. 30–42. DOI: 10.14529/cmse160203.

Введение

QR разложение прямоугольной матрицы является одним из базовых вычислительных алгоритмов для многих задач вычислительной математики. В частности, подобные вычисления возникают при решении СЛАУ, при реализации метода наименьших квадратов и задач на собственные значения [1], и т.д. Возможность эффективно параллельным образом вычислять QR разложение разреженной матрицы в некоторых случаях означает возможность использования новых классов вычислительных алгоритмов, и поэтому подобные разработки представляют практический интерес.

В работе описывается реализация на гибридной MPI+threads+SIMD архитектуре представленного в работе [2] параллельного алгоритма вычисления разреженного QR разложения для многоуровневой верхней квазитреугольной разреженной матрицы со структурой типа вложенных сечений. Алгоритм в работе [2] во многом аналогичен представленному в работах [3, 4] Тима Дэвиса с соавторами мультифронтальному алгоритму построения разреженного QR разложения. В работе [5] рассматривается многоуровневый вариант вычисления неполного разреженного QR разложения как предобусловливания при итерационном решении задачи метода наименьших квадратов.

[1] Статья рекомендована к публикации программным комитетом Международной конференции «Суперкомпьютерные дни в России – 2015».

Основные отличия предложенного в работе [2] алгоритма от предложенных ранее состоят в том, что:

- используются блочные преобразования Хаусхолдера, аналогичные представленным в работе [6];
- профильное разреженное QR разложение заменено на расширенное профильное разреженное QR разложение, которое во многих практически важных случаях дает заметно меньшее заполнение Q-фактора;
- введено дополнительное строчное упорядочивание и разбиение, которое позволяет дополнительно уменьшить связность вычислений и заполнение Q-факторов;
- предложен алгоритм построения представления матрицы, удобного для параллельного вычисления разреженного QR разложения, на основе геометрической декомпозиции расчетной области.

Данная работа, так же как и работа [2], является базовой для планируемой серии работ по новым параллельным итерационным алгоритмам решения СЛАУ и задач метода наименьших квадратов на основе композиции подпространств, порождаемых разреженными базисами. Параллельная реализация, представленная в работе, может быть взята за основу при реализации алгоритма вычисления разреженного QR разложения на гетерогенных кластерных архитектурах с ускорителями типа GPGPU.

Работа построена следующим образом. В разделе 1 приводится краткое описание параллельного алгоритма из работы [2] для построения разреженного QR разложения прямоугольной многоуровневой верхней квазитреугольной матрицы типа вложенных сечений. В разделе 2 описывается гетерогенная MPI+threads+SIMD архитектура, для которой указанный параллельный алгоритм был реализован. В разделе 3 описываются подробности реализации при отображении алгоритма на параллельную архитектуру компьютера. В разделе 4 приводится описание тестовой задачи и представлены результаты численных экспериментов. В заключении представлены выводы по результатам работы и планы по развитию и использованию представленных алгоритмов.

1. Параллельный алгоритм построения разреженного QR разложения

В этом разделе приводится краткое описание параллельного алгоритма из работы [2] для построения разреженного QR разложения прямоугольной матрицы.

Последовательный алгоритм построения QR разложения основан на блочном преобразовании Хаусхолдера вида

$$\Omega = I_M + F\,T\,F^T, \tag{1}$$

где $\Omega \in \mathbb{R}^{M\times M}$, $\Omega^T\Omega = I_M$, $F \in \mathbb{R}^{M\times s}$, $T \in \mathbb{R}^{s\times s}$. Блочное преобразование (1) строится через известный набор из s обычных векторных преобразований Хаусхолдера следующим образом: матрица F состоит из набора векторов направлений векторных преобразований Хаусхолдера, а верхняя треугольная матрица T может быть вычислена рекуррентным образом с использованием коэффициентов векторных преобразований Хаусхолдера если известна матрица $\Psi = F^TF$.

При обсуждении разреженных вычислений будут рассматриваться вопросы вычисления QR разложения для прямоугольных так называемых мелкоблочных разреженных матриц. Это означает, что разреженность понимается в смысле блоков малого размера, каждый из которых является плотной в общем случае прямоугольной матрицей. Для

простоты будем предполагать, что все мелкие блоки квадратные малого порядка s. При этом все алгоритмы могут быть обобщены на случай переменного столбцевого и строчного мелко блочного разбиения. В противовес мелким плотным $s \times s$ блокам будем говорить также о крупноблочном или просто блочном разбиении, строчном и столбцевом. Это будет означать, что соответствующие подматрицы составлены из некоторого количества мелко блочных строк и столбцов. При этом под блочным преобразованием Хаусхолдера имеется в виду преобразование вида (1) для одного мелко блочного столбца.

При последовательном построении профильного разреженного QR разложения мелко блочной разреженной матрицы C действуем по аналогии со случаем плотной матрицы. По первому мелко блочному столбцу матрицы строим разреженное блочное преобразование Хаусхолдера с разреженностью столбца такой, чтобы обнулить мелкие блоки матрицы под первой блочной диагональю. Применяем транспонированное блочное преобразование Хаусхолдера ко второму мелко блочному столбцу, для полученного результата строим следующее разреженное блочное преобразование Хаусхолдера для обнуления элементов под второй мелко блочной диагональю, и т.д.

Наравне с профильным разреженным QR разложением рассмотрим также расширенное профильное QR разложение матрицы. Схематически профильное и расширенное профильное QR изображены на рис. 1. Фактически расширенное профильное QR разложение – это профильное QR разложение, примененное к матрице, расширенной сверху нулевым квадратным блоком. При этом структура разреженности Q-фактора пополняется возможными дополнительными элементами на месте бывшего фактора R, и отсоединенными диагональными элементами, примыкающими к новому R-фактору.

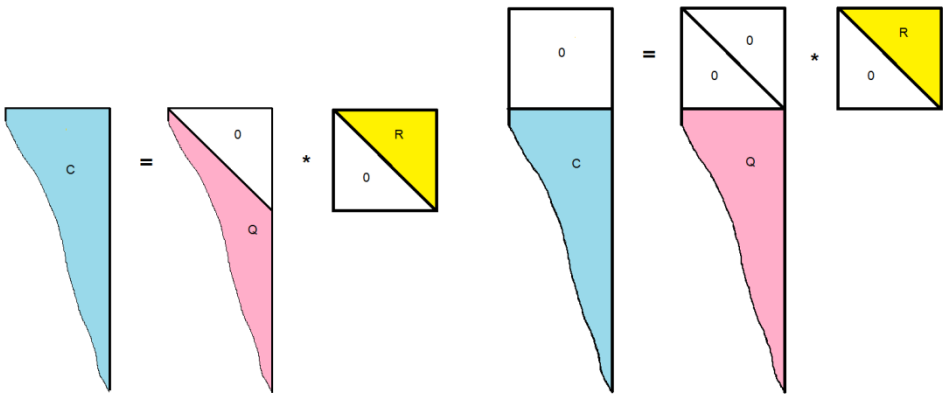

Рис. 1. Профильное (слева) и расширенное профильное (справа) QR разложения

В работе [2] показывается математическая эквивалентность профильного и расширенного профильного QR разложений в случае матрицы C полного столбцевого ранга. Также приводятся примеры, когда заполнение Q-фактора QR разложения в расширенном профильном QR разложении значительно больше заполнения Q-фактора в профильном за счет дополнительного заполнения в бывшем R-факторе разложения, а также обратный пример, представленный на рис. 2. В интересующих авторов приложениях основным является случай, когда число столбцов много меньше числа строк. В этих случаях более предпочтительным является использование варианта с расширенным

профильным разреженным QR разложением. Кроме того, использование расширенного профильного QR разложения удобно при проведении вычислений с мелко блочными матрицами, в которых число строк меньше числа столбцов.

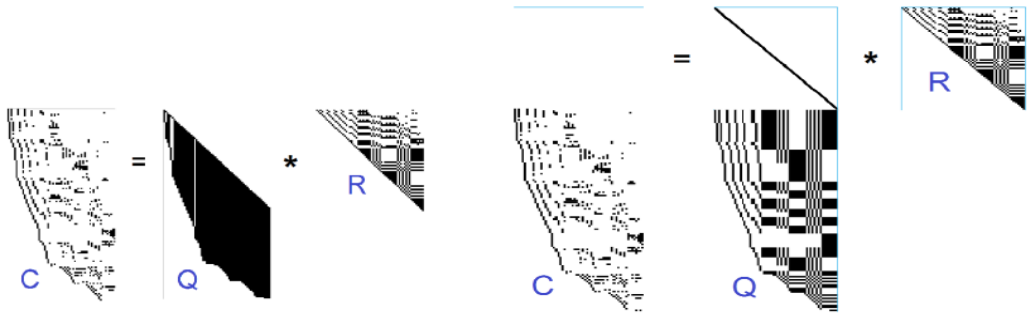

Рис. 2. Профильное (слева) и расширенное профильное (справа) QR разложения для тестовой задачи

Для прямоугольной разреженной матрицы $C \in \mathbb{R}^{M \times Ns}$ введем ее строчное разбиение в виде:

$$C = \begin{bmatrix} C_1 \\ \vdots \\ C_k \end{bmatrix}, \tag{2}$$

где $C_t \in \mathbb{R}^{M_t \times Ns}$, $t = 1, \cdots, k$, и $\sum_{t=1}^{k} M_t = M$. Пусть для каждой из матриц имеет место QR разложение с блочными преобразованиями Хаусхолдера:

$$C_t = \left(\prod_{j=1}^{N} \Omega_j{}^{(t)}\right) \begin{bmatrix} R^{(t)} \\ 0 \end{bmatrix}, \tag{3}$$

где $t = 1, \ldots, k$. Для разреженной матрицы C ее блоки C_t могут содержать много нулевых мелко блочных столбцов, в подобных случаях матрицы $R^{(t)}$ в (3) не обязательно верхние треугольные и имеют много нулевых столбцов, а среди блочных преобразований Хаусхолдера имеется много тождественных преобразований с единичной матрицей [2].

Рассмотрим задачу построения QR разложения всей матрицы (2) на основе разложений (3). Для этого рассмотрим разреженную матрицу

$$\hat{C} = \begin{bmatrix} R^{(1)} \\ \vdots \\ R^{(k)} \end{bmatrix}, \tag{4}$$

и ее QR разложение

$$\hat{C} = \left(\prod_{j=1}^{N} \hat{\Omega}_j\right) \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}, \tag{5}$$

Обозначим $\hat{\Phi} = \left(\prod_{j=1}^{N} \hat{\Omega}_j\right)$. Имеют место соотношения

$$C = \begin{bmatrix} C_1 \\ \vdots \\ C_k \end{bmatrix} = \begin{bmatrix} \Phi_1 R^{(1)} \\ \vdots \\ \Phi_k R^{(k)} \end{bmatrix} = \begin{bmatrix} \Phi_1 & & 0 \\ & \ddots & \\ 0 & & \Phi_k \end{bmatrix} \hat{\Phi} \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix} = \Phi \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}, \tag{6}$$

где

$$\Phi = \begin{bmatrix} \Phi_1 & & 0 \\ & \ddots & \\ 0 & & \Phi_k \end{bmatrix} \hat{\Phi}, \tag{7}$$

причем $\Phi^T\Phi = I_M$. Отсюда следует, что неявное представление (7) совместно с равенством $C = \Phi\begin{bmatrix}\hat{R}\\0\end{bmatrix}$ из (6) есть QR разложение матрицы C, при этом матрица Φ представляет собой Q-фактор QR разложения, а квадратная верхняя треугольная матрица $\hat{R}$ есть R-фактор QR разложения.

Описанная конструкция, очевидно, позволяет параллельным образом вычислять QR разложение матрицы за счет введения строчного разбиения, поскольку строчные QR разложения (3) можно считать независимо. Синхронизация вычислений происходит только при вычислении объединяющего QR разложения (5). С другой стороны понятно, что подобный подход к основному распараллеливанию вычислений может быть эффективен только если число столбцов в матрице существенно меньше числа строк, иначе затраты на объединяющее QR разложение могут доминировать.

Описанный параллельный алгоритм вычисления QR разложения по блочным строкам можно сделать более эффективным за счет использования дополнительной столбцевой разреженности матрицы. Для этого рассмотрим двухуровневую организацию вычислений для прямоугольной матрицы C со структурой разреженности, изображенной на рис. 3. Пусть число мелко блочных столбцов в матрицах C_1, C_2 и C_S равны соответ-

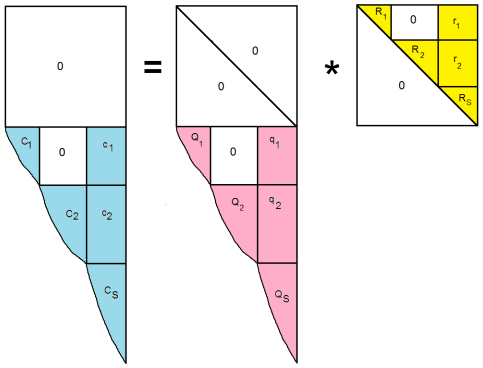

Рис. 3. Двухуровневая организация параллельного вычисления QR разложения

ственно N_1, N_2 и N_S, где $N_1 + N_2 + N_S = N$. Как показано в [2], для матрицы C с такой структурой разреженности для соответствующей матрицы $\hat{C}$ типа (4) в объединяющем QR разложении задачу вычисления ее QR разложения можно перестановкой блочных строк свести к задаче вычисления QR разложения с разреженной матрицей мелко блочного размера $(n_1 + n_2 + N_S) \times N_S$, здесь n_1 и n_2 соответственно число ненулевых мелко блочных столбцов в матрицах c_1 и c_2.

Определение 1. Квадратную матрицу с заданным в ней строчным и столбцевым разбиением и имеющую блочную структуру разреженности вида:

$$\begin{bmatrix} A_1 & 0 & a_1 \\ 0 & A_2 & a_2 \\ b_1 & b_2 & D \end{bmatrix} \tag{8}$$

будем называть матрицей *со структурой разреженности типа вложенных сечений.*

Квадратную матрицу (8) естественно ассоциировать с двухуровневым бинарным деревом, в котором корневой узел бинарного дерева соответствует блоку окаймления, а листья дерева соответствуют диагональным блокам A_1 и A_2. По этой причине матрицу в (8) будем считать двухуровневой матрицей со структурой разреженности типа вло-

женных сечений. Если диагональные блоки A_1 и A_2 также являются матрицами со структурой разреженности типа вложенных сечений, то такой матрице можно сопоставить трехуровневое бинарное дерево и рассматривать всю матрицу в совокупности как трехуровневую, т.д. Алгоритмы вычисления упорядочивания матрицы, приводящие ее к виду (8), детально рассмотрены в [7].

Обобщая этот подход на случай прямоугольных матриц введем следующее определение [2].

Определение 2. Прямоугольную мелко блочную матрицу с введенным в ней блочным строчным и столбцевым разбиениями будем называть *верхней квазитреугольной - уровневой матрицей со структурой разреженности типа вложенных сечений*, если в терминах крупных блоков матрица является квадратной верхней треугольной и имеет структуру разреженности типа вложенных сечений, описываемой L-уровневым деревом зависимостей.

В частности, матрица на рис. 3 в терминах Определения 1 является двухуровневой верхней квазитреугольной с двухуровневым бинарным деревом зависимостей вычислений.

Как следует из предыдущего изложения, для эффективного вычисления QR разложения верхних квазитреугольных матриц с разреженностью типа вложенных сечений можно использовать следующий параллельный алгоритм:

1. Параллельно и независимо для каждой блочной строки матрицы строим расширенное профильное разреженное QR разложение на основе блочных преобразований Хаусхолдера.
2. Параллельно в порядке, определяемом деревом зависимостей вычислений, достраиваем QR разложения для объединяющих подматриц для вычисления QR разложений соответствующих мелко блочных столбцевых окаймлений.

2. Архитектура гибридной вычислительной системы

Большинство современных суперкомпьютерных вычислительных систем, как правило, имеют неоднородную архитектуру. С одной стороны, имеется набор вычислительных узлов с распределенной памятью, обмен данными между которыми может быть осуществлен по быстрой обменной сети. С другой стороны, каждый узел представляет со-

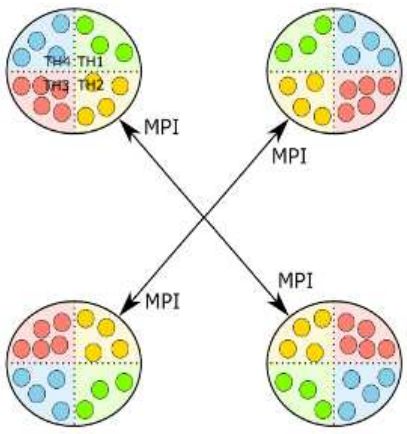

Рис. 4. Двухуровневая MPI+TBB организация параллельных вычислений

бой многопроцессорный/многоядерный компьютер с общим доступом к оперативной памяти. Программная реализация вычислительных алгоритмов (включая алгоритм вычисления разреженного QR разложения) на компьютерах подобной архитектуры предполагает использование стандарта MPI при распараллеливании по распределенной памяти (по узлам вычислений), а по общей памяти узла распараллеливание по процессорам/ядрам естественно осуществлять на основе стандартов работы с потоками с встроенными механизмами динамической балансировки нагрузки, таких как OpenMP или Intel Threading Building Blocks (TBB). При этом предполагается (рис. 4), что на каждом узле имеется только один MPI процесс, который порождает на этом узле нужное количество одновременно работающих потоков вычислений.

Большинство современных процессоров для оптимизации времени выполнения кода поддерживают так называемые векторные расширения систем команд — SIMD (Single Instruction Multiple Data) инструкции. Подобные вычисления можно проводить на любом ядре центрального процессора. В этих расширениях вычисления осуществляются с векторами данных стандартного целого и вещественного типа. Любое вычисление на векторных регистрах осуществляется в следующей последовательности: данные из памяти загружаются в регистровые переменные, производится вызов аппаратно поддерживаемой функции работы с регистрами, затем данные обратно выгружаются в обычную память. Выпускаемые сейчас процессоры обычно поддерживают системы команд SSE и AVX, работающие соответственно со 128-битными XMM и 256-битными YMM регистрами. Это позволяет при использовании 256-битных регистров YMM, например, за один такт сложить 8 чисел с плавающей точкой одинарной точности или перемножить 4 числа с двойной точностью. На некоторых новейших процессорах поддерживается система команд AVX2, в которой дополнительно по отношению к AVX имеются FMA команды, совмещающие сложение и умножение векторов. В ускорителях Intel Xeon Phi имеется поддержка 512-битных ZMM регистров. В следующем поколении ускорителей Knights Landing появится аппаратная поддержка новой системы команд AVX512, совместимой с серверными процессорами Xeon.

3. Отображение алгоритма разреженного QR разложения на архитектуру вычислительной системы

Приведенный во втором разделе параллельный алгоритм вычисления разреженного QR разложения для верхней квазитреугольной матрицы типа вложенных сечений был реализован на кластерной MPI+threads архитектуре с использованием SIMD инструкций. Распараллеливание алгоритма на гетерогенной MPI+threads+SIMD архитектуре осуществлено следующим образом.

Распараллеливание верхнего уровня по MPI осуществлялось как распараллеливание по распределенной памяти. Для этих целей в дереве зависимостей вычислений каждому MPI процессу было выделено целиком поддерево зависимых вычислений по возможности с близкой для всех поддеревьев вычислительной работой. Дополнительная динамическая балансировка вычислений в какой-либо форме не проводилась. Обработка каждого из вышестоящих узлов дерева зависимостей передавалась одному (например первому) из тех процессоров, который обрабатывал один из узлов сыновей данного узла. На каждый MPI процесс перераспределялись те блочные строки матрицы, которые нужны для окончательной обработки своих узлов поддеревьев зависимостей вычислений.

Распараллеливание среднего уровня по нитям осуществлялось с помощью технологии Intel TBB либо как независимые, либо как зависимые вычисления. Зависимости описываются в виде подграфа зависимых вычислений для узлов поддеревьев своего MPI процессора, независимые вычисления проводились при начальном вычислении QR разложений для блочных строк. При проведении вычислений с узлом дерева зависимых вычислений, не входящим в MPI-поддеревья, вычисление объединяющих QR разложений проводилось только при поступлении необходимых данных с других MPI процессов.

Распараллеливание нижнего уровня параллельных вычислений — SIMD векторизация — проводилось за счет использования блочных преобразований Хаусхолдера. Рассмотрим этот вопрос подробнее.

Основными операциями при работе с блочными преобразованиями Хаусхолдера являются:

1. Вычисление векторного QR разложения с векторными преобразованиями Хаусхолдера для мелко блочного столбца.
2. Преобразование набора векторных преобразований Хаусхолдера в единое блочное преобразование Хаусхолдера для мелко блочного столбца.
3. Применение с учетом разреженности блочных преобразований Хаусхолдера к последующим мелко блочным столбцам матрицы.

С учетом сказанного в разделе 1 про способ трансформации векторных преобразований Хаусхолдера в блочное можно выделить следующие 4 основные вычислительные операции в терминологии функций BLAS 1:

- операция DOT: $z = x^T y$ — вычисление скалярного произведения векторов;
- операция AXPY: $y := y + ax$ — прибавление масштабированного вектора;
- блочное обобщение операции DOT: $Z = X^T Y$ — скалярное произведение для блоков векторов;
- блочное обобщение операции AXPY: $Y := Y + XA$ — прибавление блока векторов умноженного на квадратную матрицу.

Векторные операции встречаются при векторном вычислении QR разложения. В операциях DOT и AXPY длины векторов недостаточны для покрытия накладных расходов вызова оптимизированных BLAS функций из Intel MKL. По этой причине в этих операциях осуществлялась ручная SIMD векторизация прямым вызовом соответствующих векторных инструкций с помощью интринсик функций.

Для максимальной локализации работы с памятью при обработке блочных преобразований Хаусхолдера естественно использовать формат хранения данных «по строкам» вместо традиционно используемого для блока векторов формата «по столбцам».

При проведении SIMD векторизации для блоков векторов в силу особенностей векторных инструкций реализовывалась поддержка только значений $s = 2; 4; 8; 16$ для двух типов данных float и double. Как уже отмечалось, разреженные QR разложения будут использоваться в контексте построения разреженных базисов в алгоритмах решения СЛАУ и для реализации метода наименьших квадратов, а в этом случае параметр s — это число векторов в блоке одновременно обрабатываемых в итерационной схемы. Для длинных блоков векторов задача вычисления блочных операций DOT и AXPY сводилась к циклу вызовов для подматриц размера $s \times s$. Подробности реализации этих операций в терминах SIMD инструкций для подматриц стандартного размера можно найти в работе [8]. Как показали численные эксперименты, ручная векторизация для таких

маленьких порядков s оказалась значительно эффективней вызовов библиотечных реализаций из Intel MKL и Intel IPP.

4. Результаты численных экспериментов

Для тестирования предложенных в работе алгоритмов был выбран искусственный тест, в котором по возможности отражены основные особенности будущего использования алгоритмов.

Для регулярной $N_x \times N_y \times N_z$ прямоугольной сетки была построена регулярная разреженная мелко блочная матрица с блоками порядка $s = 8$, структура разреженности которой отвечает шаблону уравнения Лапласа. Выбор порядка блока был обусловлен тем, что для такого значения порядка блока для типов данных float и double возможно добиться эффективного использования SIMD векторных инструкций вплоть до набора AVX2.

Для полученной матрицы с помощью алгоритма вложенных сечений, как описано в работе [2], было построено упорядочивание и разбиения, приводящие матрицу к виду -уровневой верхней квазитреугольной матрицы типа вложенных сечений. Обозначим эту мелко блочную матрицу A. Для этой матрицы был построен разреженный блочно-диагональный ортонормированный базис P. Разреженный базис P является мелко блочной матрицей, строчный и столбцевой размеры каждого мелкого блока равны s и совпадают с порядком мелкого блока матрицы A.

Матрица P представляет собой блочно-диагональную матрицу вида

$$P = \begin{bmatrix} P_1 & & 0 \\ & \ddots & \\ 0 & & P_l \end{bmatrix},$$

где l — число крупных блоков в столбцевом разбиении матрицы A. Очевидно, что матрица C такая, что

$$C = AP,$$

также как и A, представляет собой мелко блочную матрицу с блоками порядка s. Кроме того, матрица C является также -уровневой верхней квазитреугольной матрицей типа вложенных сечений, и число мелко блочных столбцов в ней есть полное число мелко блочных столбцов в базисе P.

Эксперименты по вычислению разреженного QR разложения проводились для построенной таким образом матрицы C. В частности, если число столбцов в базисе P много меньше порядка матрицы A, то в основных строчных подматрицах разреженного QR разложения число столбцов много меньше числа строк, что оправдывает использование расширенного разреженного QR разложения в основных вычислениях. Кроме того, каждый блок P_j базиса обладает внутренней разреженностью, а потому результат произведения также есть разреженная мелко блочная матрица.

Однопроцессорное тестирование алгоритма проводилось на самом современном 18-ядерном процессоре Intel Xeon E5-2699v3 с архитектурой Haswell под управлением CentOS 6.6. Для тестирования алгоритма была построена матрица с числом строк около 1,5 млн. и порядком мелкого блока $s = 8$ и количеством столбцов около 0,1 млн.

Тестовое приложение компилировалось с помощью оптимизирующего компилятора ICC-15.0.3. В табл. 1 и 2 представлены времена работы алгоритма для матрицы с оди-

нарной и двойной точностью соответственно для различных наборов векторных инструкций и количества потоков.

Таблица 1

Время работы алгоритма с числами одинарной точности (с)

arch\threads	1	2	4	8	12	16	18
no-vec	2,197	1,111	0,609	0,356	0,254	0,197	0,178
SSE	1,122	0,570	0,312	0,184	0,131	0,104	0,092
AVX	0,887	0,453	0,245	0,143	0,103	0,082	0,076
AVX2	0,711	0,367	0,196	0,119	0,086	0,068	0,062

Таблица 2

Время работы алгоритма с числами двойной точности (с)

arch\threads	1	2	4	8	12	16	18
no-vec	2,447	1,235	0,677	0,386	0,276	0,216	0,198
SSE	1,985	1,010	0,555	0,316	0,224	0,176	0,162
AVX	1,466	0,743	0,399	0,231	0,165	0,134	0,120
AVX2	1,043	0,528	0,282	0,166	0,120	0,096	0,090

Из результатов видно, что использование самых современных векторных инструкций позволяет получить ускорение до 3 раз по сравнению с оптимизирующим компилятором ICC. Ускорение по сравнению с бесплатным компилятором GCC получается еще более значительным. Использование всех 18 ядер процессора ускоряет работу алгоритма в обоих случаях в более чем 11 раз. На рис. 5 изображен профиль загрузки потоков в тестовом приложении, полученном с помощью программы Intel VTune Amplifier, оранжевым цветом отмечены регионы синхронизации потоков. Первая оранжевая область на временной линии отвечает окончанию вычисления независимых QR разложений для

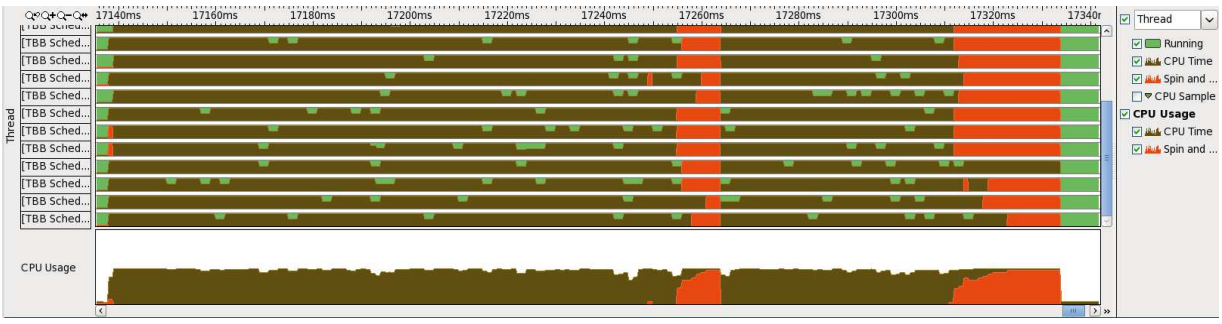

Рис. 5. Профиль загрузки потоков в тестовом приложении

крупных блочных строк, а вторая — окончанию обработки зависимых вычислений по дереву. Видно, что при обработке зависимых блоков все еще остается значительный дисбаланс загрузки ядер, что не позволяет вплотную приблизиться к линейной масштабируемости вычислений. В дальнейшем планируется уделить больше внимания этим ме-

стам в алгоритме. Таким образом, использование обоих механизмов распараллеливания в современных процессорах позволяет достичь ускорения в более чем 30 раз.

Тестирование в гибридном параллельной MPI+threads+SIMD режиме проводилось на суперкомпьютере Ломоносов-2, каждый узел которого содержит 10-ти ядерный процессор Intel Xeon E5-2680v2 с архитектурой Ivy Bridge, процессоры поддерживают технологию AVX. Для тестирования алгоритма была построена матрица с числом строк равным порядка 4,7 млн., порядком мелкого блока $s = 8$ и количеством столбцов около 0,6 млн. Результаты экспериментов представлены в табл. 3 и 4, каждый MPI процесс использовал все 10 ядер процессора в многонитевом режиме с использованием технологии AVX.

Таблица 3

Время работы алгоритма с числами одинарной точности (с)

Nproc	**1**	**2**	**4**	**8**	**16**	**32**
Время	0,160	0,086	0,035	0,022	0,015	0,008

Таблица 4

Время работы алгоритма с числами двойной точности (с)

Nproc	**1**	**2**	**4**	**8**	**16**	**32**
Время	0,240	0,125	0,064	0,035	0,022	0,012

Заключение

В работе представлена реализация на гибридной параллельной MPI+threads+SIMD архитектуре параллельного алгоритма вычисления QR разложения многоуровневой разреженной верхней квазитреугольной матрицы со структурой разреженности типа вложенных сечений. Результаты численных экспериментов с предложенным алгоритмом для тестовых задач на гибридной параллельной MPI+threads+SIMD архитектуре показывают высокую эффективность предложенных алгоритмов: ускорение до 3 раз от использования векторных инструкций AVX2, ускорение до 11 раз при использовании 18 ядер процессора, ускорение до 20 раз при использовании 32 процессоров. Результаты также показывают, что float вычисления по сравнению с double кроме двукратной экономии памяти дают также ускорение вычислений в 1,5 раза. Результаты работы планируется использовать при реализации массивно-параллельных алгоритмов решения СЛАУ на основе композиции подпространств, порождаемых разреженными базисами. Также планируется развитие алгоритмов в направлении использования ускорителей типа GPGPU.

Литература

1. Тыртышников Е.Е. Методы численного анализа: учеб. пособие для студ. вузов. М.: Издательский центр «Академия», 2007. 320 с.
2. Харченко С.А. Параллельный алгоритм разреженного QR разложения для прямоугольных верхних квазитреугольных матриц со структурой типа вложенных сечений // Вычислительные методы и программирование. 2015. Т. 16. С. 566–577.

3. Davis T.A. Algorithm 915: SuiteSparseQR, a multifrontal multithreaded sparse QR factorization package // ACM Trans. Math. Softw. Dec. 2011 Vol. 38, No. 1 P. 8:1–8:22.
4. Yeralan S.N., Davis T.A., Ranka S. Algorithm 9xx: Sparse QR Factorization on the GPU // ACM Transactions on Mathematical Software. Jan. 2015. Vol. 1, No. 1, Article 1. P. 1–28.
5. Rotella F., Zambettakis I. Block Householder transformation for parallel QR factorization // Appl. Math. Letters. Vol. 12, I. 4. 1999. P. 29–34.
6. Li N., Saad Y. MIQR: A multilevel incomplete QR preconditioner for large sparse least-squares problems // SIAM. J. Matrix Anal. Appl. 28(2). 2006. P. 524–550.
7. George A., Liu J.W. Computer Solution of Large Sparse Positive Definite Systems. Prentice Hall, 1981. 324 p.
8. Андреев А.Е., Егунов В.А., Насонов А.А., Новокщенов А.А. Применение векторных инструкций в алгоритмах блочных операций линейной алгебры // Известия ВолгГТУ. Серия: Актуальные проблемы управления, вычислительной техники и информатики в технических системах. 2014. № 12 (139). С. 5–11.

Харченко Сергей Александрович, инженер, ООО «ТЕСИС» (Москва, Российская Федерация), skh@tesis.com.ru.

Ющенко Алексей Александрович, инженер, ООО «ТЕСИС» (Москва, Российская Федерация), ay@tesis.com.ru.

Поступила в редакцию 16 декабря 2015 г.

Bulletin of the South Ural State University
Series "Computational Mathematics and Software Engineering"
2016, vol. 5, no. 2, pp. 30–42

DOI: 10.14529/cmse160203

PARALLEL IMPLEMENTATION OF THE SPARSE QR DECOMPOSITION FOR RECTANGULAR UPPER QUASI TRIANGULAR MATRIX WITH ND-TYPE SPARSITY

S.A. Kharchenko, LLC «TESIS», Moscow, Russian Federation
A.A. Yushchenko, LLC «TESIS», Moscow, Russian Federation

The paper considers parallel MPI+threads+SIMD implementation of the algorithm for computing sparse QR decomposition of a specially ordered rectangular matrix. Decomposition is based on block sparse Householder transformations. The algorithm starts with independent parallel QR decompositions for sets of matrix rows; and then, according to the computations tree, the QR decomposition is performed for matrices, combined with elements of R factors of rows decompositions. The results of numerical experiments for test problems show efficiency of the parallel implementation. The algorithm can also be efficiently implemented on heterogeneous cluster architectures with GPGPU accelerators.

Keywords: sparse rectangular matrix, upper quasi triangular matrix, nested dissection, QR decomposition, Householder transformations, MPI, multithreading, SIMD.

FOR CITATION

Kharchenko S.A., Yushchenko A.A. Parallel Implementation of the Sparse QR Decomposition for Rectangular Upper Quasi Triangular Matrix with ND-Type Sparsity. *Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering.* 2016. vol. 5, no. 2. pp. 30–42. (in Russian) DOI: 10.14529/cmse160203.

References

1. Tyrtyshnikov E. E. *Metody chislennogo analiza* [Methods of Numerical Analysis]. Akademiya, Moscow, 2007. 320 p. (in Russian)
2. Kharchenko S.A. Parallel'nyy algoritm razrezhennogo QR razlozheniya dlya pryamougol'nykh verkhnikh kvazitreugol'nykh matrits so strukturoy tipa vlozhennykh secheniy [A Parallel Algorithm for the Sparse QR Decomposition of a Rectangular Upper Quasi-Triangular Matrix with ND-Type Sparsity]. *Vychislitel'nye metody i programmirovanie* [Numerical Methods and Programming]. 2015. vol. 16. pp. 566–577. (in Russian)
3. Davis T.A. Algorithm 915: SuiteSparseQR, a Multifrontal Multithreaded Sparse QR Factorization Package. ACM Trans. Math. Softw. Dec. 2011. vol. 38, no. 1. pp. 8:1–8:22.
4. Yeralan S.N., Davis T.A., Ranka S. Algorithm 9xx: Sparse QR Factorization on the GPU. ACM Transactions on Mathematical Software. Jan. 2015. vol. 1, no. 1, Article 1. pp. 1–28.
5. Rotella F., Zambettakis I. Block Householder Transformation for Parallel QR Factorization. Appl. Math. Letters. 1999. vol. 12, i. 4. pp. 29–34.
6. Li N., Saad Y. MIQR: A Multilevel Incomplete QR Preconditioner for Large Sparse Least-Squares Problems. SIAM. J. Matrix Anal. Appl. 2006. vol. 28(2). pp. 524–550.
7. George A., Liu J. W. Computer Solution of Large Sparse Positive Definite Systems. Prentice Hall. 1981. 324 p.
8. Andreev A.E., Egunov V.A., Nasonov A.A., Novokshenov A.A. Primenenie vektornykh instruktsiy v algoritmakh blochnykh operatsiy lineynoy algebry [Application of vector instructions in algorithms of block operations of linear algebra]. *Izvestiya VolgGTU. Seriya: Aktual'nye problemy upravleniya, vychislitel'noy tekhniki i informatiki v tekhnicheskikh sistemakh* [VSTU News: "Actual Problems of Control, Computers and Informatics in Technical Systems"]. Volgograd. 2014. vol. 39(12). pp. 5–11. (in Russian)

Received December 16, 2015.