ПРИМЕНЕНИЕ ТЕХНОЛОГИЙ ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ NVIDIA CUDA ДЛЯ ПОИСКА СОБСТВЕННЫХ ЗНАЧЕНИЙ МАТРИЦ

А.В. Высоцкий, Н.Е. Тимофеева, А.Н. Савин, К.И. Шоломов

Саратовкский государственный университет ин. Н. Г. Чернышевского

Введение

В настоящее время широко применяется математическое моделирование различных динамических систем, описываемых дифференциальными уравнениями. При этом приходится часто решать задачу связанную с поиском собственных значений матриц (например, при поиске собственных частот колебаний динамической системы). Одной из важных проблем, возникающих при этом, является большие вычислительные затраты, а следовательно, длительное время вычислений.

Существует большое количество методов, позволяющих решать проблему поиска собственных значений [1]. Нахождение собственных значений матриц, имеющих размерность порядка 10^3 - 10^6 и более, требует больших вычислительных затрат, а следовательно, длительное время вычислений. Но с другой стороны задаче нахождения собственных значений присущ внутренний параллелизм, что дает основания использовать параллельные вычислительные системы для их решения.

Одной из платформ, ориентированной для осуществления различных матричных преобразований, является программно-аппаратная платформа NVIDIA CUDA. Современные графические ускорители NVIDIA(GPU) имеют сотни упрощенных вычислительных ядер и большие объемы встроенной памяти, позволяющие максимально эффективно решать задачи, в которых необходимо одинаковые операции применить к множеству независимых данных[2].

Аппаратные возможности графических процессоров NVIDIA в совокупности с поставляемой с ними библиотекой CUBLAS – набором подпрограмм, предназначенных для вычислений задач линейной алгебры и использующие прямой доступ к ресурсам GPU, можно использовать для поиска собственных значений у матриц высоких порядков.

Целью работы являются изучение технологии параллельных вычислений NVIDIA CUDA и реализация различных методов поиска собственных значений с помощью средств библиотеки CUBLAS и оценка их эффективности.

Постановка задачи

Пусть L – линейное пространство над полем K, $A: L \to L$ – линейное преобразование.

Собственным значением линейного преобразования A называется такое число $\lambda \in K$, что для некоторого ненулевого вектора $x \in L$ имеет место равенство:

$$Ax = \lambda x.$$

Любой ненулевой вектор x, удовлетворяющий данному равенству, называется собственным вектором матрицы A, соответствующим собственному значению λ.

Условием существования собственных значений матрицы является требование:

$$|A - \lambda E| = 0,$$

где E – единичный оператор.

Данный определитель называют характеристическим уравнением оператора A, а матрицу $A - \lambda E$ - характеристической матрицей:

$$A - \lambda E = \begin{pmatrix} a_{11} - \lambda & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} - \lambda \end{pmatrix}$$

$$|A - \lambda E| = \begin{vmatrix} a_{11} - \lambda & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} - \lambda \end{vmatrix}$$

Задача нахождения собственных значений матрицы состоит в решении характеристического уравнения, т.е. нахождения корней характеристического многочлена:

$$\Delta_A(\lambda) = \det(A - \lambda E) = a_n(\lambda - \lambda_1)^{n_1}(\lambda - \lambda_2)^{n_2} \dots (\lambda - \lambda_k)^{n_k}$$

где

$\lambda_1, \lambda_2, \dots, \lambda_k$ - корни многочлена кратности $n_1, n_2, \dots, n_k$, причем $\sum_i n_i = n$.

Поиск собственных значений, используя разложение Холецкого

Разложение Холецкого — представление симметричной положительно-определённой матрицы A в виде $A = LL^T$, где L — нижняя треугольная матрица со строго положительными элементами на диагонали. Разложение Холецкого всегда существует и единственно для любой симметричной положительно-определённой матрицы [3].

Метод разложения Холецкого можно применить для нахождения A_k собственных значений. Для этого для матрицы A необходимо построить последовательность $\{A_k\}$ матриц по следующим правилам:

$$A_1 = A;$$

Для всех $k = 1, 2, \dots$ матрица A_{k+1} получается из матрицы следующим образом:

1. Строим разложение Холецкого матрицы A_k: $A_k = L_k L_k^T$,
2. Вычисляем матрицу A_{k+1} как произведение матриц L_k^T и L_k: $A_{k+1} = L_k^T L_k$

Алгоритм поиска собственных значений симметричной, положительно определенной матрицы A заключается в следующем:

1. Осуществляем разложение Холецкого матрицы A: $A = LL^T$;
2. Получаем новое значение матрицы A путем умножения матрицы L^T на L : $A' = L^T L$;
3. Если сумма внедиагональных элементов для каждого столбца и строки больше заданного ε, возвращаемся к шагу 1;
4. Возвращаем диагональные элементы матрицы A в качестве результата вычислений [3].

Наиболее эффективное распараллеливание приведенного выше алгоритма можно получить, распараллелив следующие функции:

- поиск элементов в первом столбце при разложении матрицы методом Холецкого;
- вычисление элементов строки при разложении матрицы методом Холецкого;
- перемножение матриц.

Для распараллеливания разложения Холецкого создается двумерный набор потоков размерности равной порядку матрицы. В каждом потоке осуществляется обработка соответствующего ему элемента матрицы. Таким образом, в параллельных участках кода, достигается ускорение вычислений в число равное порядку матрицы.

Перемножение матриц осуществляется за счет вызова функции cublasSgemm_v2 из библиотеки CUDA BLAS. Данная функция имеет сложную сигнатуру, поэтому в качестве ряда параметров передавались определенные значения для того, чтобы функция осуществляла перемножение матрицы на ее транспонированное значение.

После вызова каждого блока вычислений на графических процессорах, осуществляется синхронизация поток для гарантирования завершения вычислений на всех потоках.

Блок-схема данного алгоритма приведена на рисунке 1.

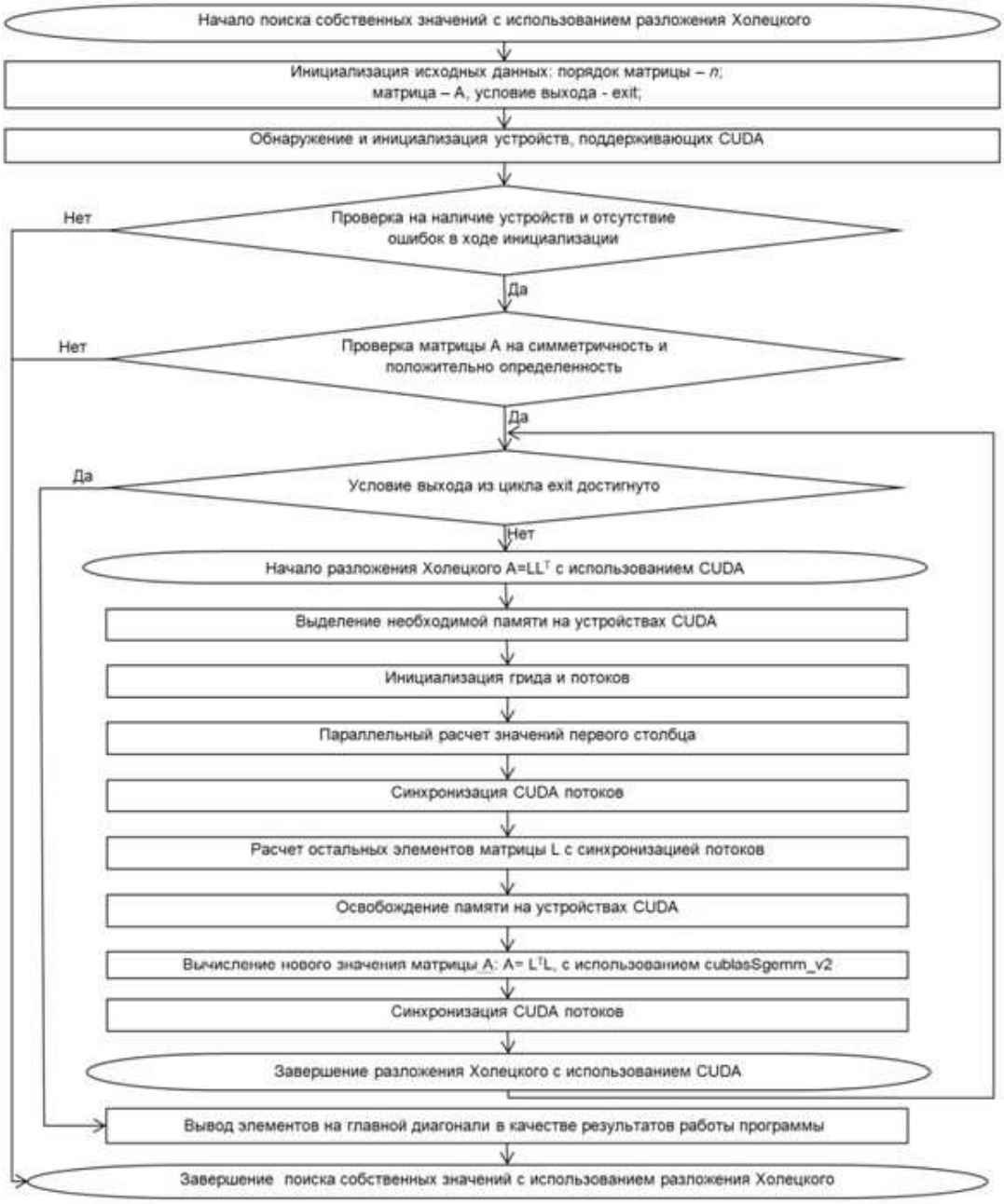

Рис. 1. Блок-схема параллельного алгоритма поиска поиска собственных значений матрицы с применением разложения Холецкого

Поиск собственных значений, используя метод Хаусхолдера

Одним из возможных подходов к нахождению собственных значений матрицы является использование метода Хаусхолдера, приводящего исходную симметричную матрицу к трехдиагональной симметричной форме, вместе с QL-алгоритом . Метод Хаусхолдера заключается в умножении матрицы на каждом шаге справа и слева на ортогональные матрицы отражения. Каждое преобразование обнуляет часть строки и столбца в преобразовываемой матрице. В QL-алгоритме используются двусторонние умножения на элементарные матрицы вращения.

Рассмотрим квадратную симметричную матрицу $A = (a_{ij})$ размерности n. Метод Хаусхолдера позволяет привести матрицу A к трехдиагональной форме за $n - 2$ ортогональных преобразований. Преобразования осуществляется с использованием матрицы Хаусхолдера, имеющей следующий вид [4]:

$$H = E - 2\frac{vv^t}{v^t v}, \quad (1)$$

где v – произвольный ненулевой вектор-столбец, E – единичная матрица, vv^T - квадратная матрица того же размера. Положим $A_0 = A$. Для обнуления $n - 2$ элементов первого столбца

$b = (a_{11}^0, a_{21}^0, \dots, a_{n1}^0)$ вектор v составляется из нижних $n-1$ элементов данного столбца следующим образом:

$$v = b + \mathrm{sign}(b_1)||b||e, \quad (2)$$

где $||b|| = \left(\sum_{i=1}^{n} a_{i1}^2\right)^{1/2}$ – эвклидова норма вектора, $e = (1,0,\dots,0)^T$, и дополняется нулевыми элементами до размерности n, добавленных в начало вектора.

Обнуление элементов первого столбца и строки матрицы A происходит во время следующего преобразования:

$$A_1 = HA_0H. \quad (3)$$

Алгоритм тридиагонализации симметричной квадратной матрицы приведен ниже.

На k -м шаге выполняются следующие действия:

1. Из -го столбца b матрицы выбираются $n-k$ нижних элементов;
2. Составляется вектор v на основе вектора b по формуле 1;
3. Расширить вектор v до размерности n нулевыми элементами
4. Вычисляется матрица Хаусхолдера H по формуле 2;
5. Производится умножение преобразуемой матрицы справа и слева на вычисленную матрицу H

При распараллеливании метода Хаусхолдера возникает необходимость взаимодействия процессов, осуществляющих ортогонализацию, что достаточно сложно реализовать на CUDA, где фактически отсутствует возможность какого-либо взаимодействия между исполняемыми процессами. Повышение эффективности метода достигается за счет использования подпрограмм CUBLAS, реализующих матрично-векторные преобразования на GPU, и параллельной обработки массивов данных.

На рисунке 2 приведена блок-схема тридиагонализации матрицы, используя метод Хаусхолдера.

Рис. 2. Блок-схема параллельного алгоритма поиска поиска собственных значений матрицы с применением метода Хаусхолдера

Цепочка $n-2$ таких преобразований приводит к исходную матрицу A к трехдиагональному виду. Поиск собственных значений, после проведенных преобразований, осуществляется с помощью QL-алгоритма с неявными сдвигами, который позволяет за небольшое число шагов заданной точностью вычислить собственные значения трехдиагональных матриц [5].

Результаты численных экспериментов

Описанные методы были реализованы на языке C++ с использованием библиотеки CUBLAS. На рисунке 3 приведено сравнение усредненного по итогам 50 запусков времени поиска собственных значений методом Холецкого и методом Хаусхолдера. Вычисления проводились на GPU NVIDIA GeForce GTX 650 2Gb.

Из представленного графика видно, что оба метода показывают практически одинаковое время вычислений. Метод Хаусхолдера демонстрирует незначительное преимущество перед методом Холецкого для матриц размерности до $n = 4000$. При увеличении размерности наблюдается последовательное увеличение эффективности метода Холецкого. Это можно объяснить наличием ветвлений в методе Хаусхолдера и большом количестве последовательных действий в QL-алгоритме, что с увеличением размерности преобразовываемых матриц сказывается на эффективности.

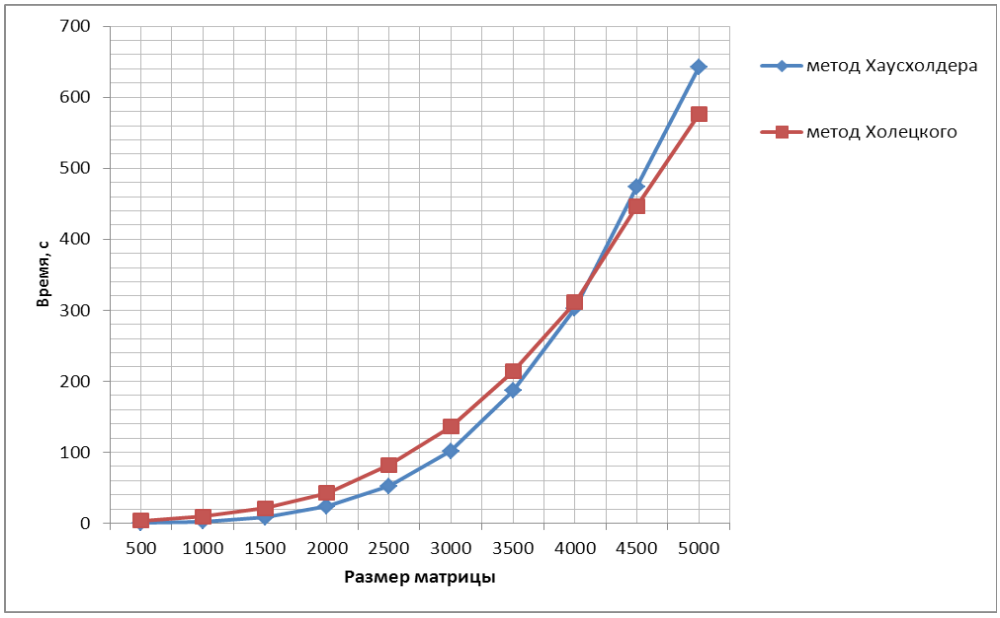

Рис. 3. Зависимость времени (в секундах) выполнения поиска собственных значений симметричной матрицы от ее размера.

На рисунке 4 продемонстрировано усредненное время поиска собственных значений последовательными и параллельными версиями методов.

Полученные результаты показывают, что использование GPU в вычислениях дает значительный прирост производительности по сравнению с последовательными реализациями.

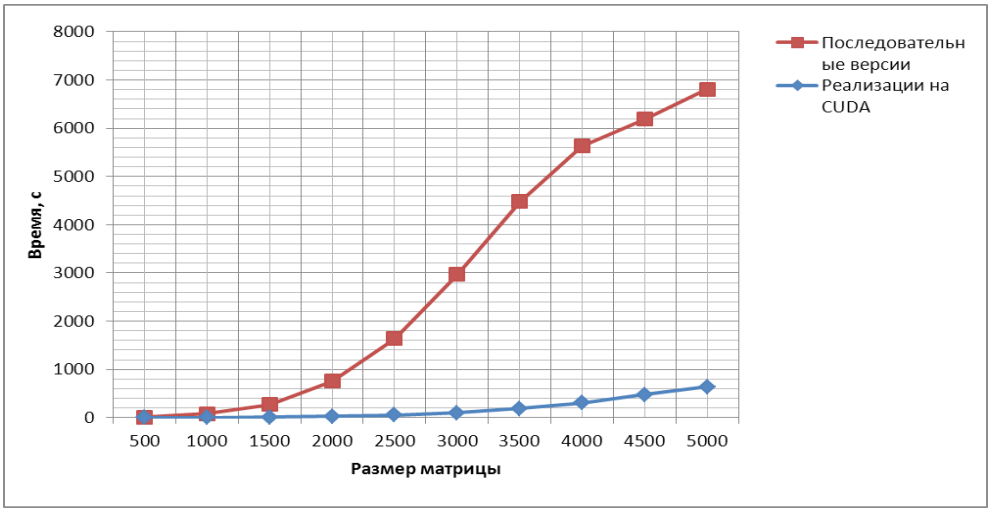

Рис. 4. Зависимость времени (в секундах) выполнения поиска собственных значений последовательными и распараллеленными реализациями

Поиска собственных значений методом Хаусхолдера также был реализован в среде графического программирования LabView в виде отдельного модуля, позволяющего провести сравнительный анализ производительности реализованного модуля, штатных средств LabView и Multicore Analysis and Sparse Matrix Toolkit. На рисунке 5 представлена лицевая панель модуля.

а б

в

Рис. 5. Лицевая панель модуля поиска собственных значений в среде графического программирования LabView: *а* – штатные средства LabView, *б* – метод Хаусхолдера, *в* – модуль, реализованный с использованием Intel MKL

По итогам тестирования производительности, модуль на CUDA продемонстрировал небольшое преимущество перед реализацией на Intel MKL. Данный модуль предназначен для интеграции в программный комплекс расчета атомно-молекулярных структур квантово-механическим методом.

Заключение

В рамках данной работы была изучена технологии параллельных вычислений NVIDIA CUDA и реализованы различные методы поиска собственных значений с помощью средств библиотеки CUBLAS и проведен анализ их эффективности.

ЛИТЕРАТУРА:

1. Б.Парлетт Симметричная проблема собственных значений. Численные методы. Москва «Мир», 1983 г. С. 156.
2. А.В. Боресков Параллельные вычисления на GPU. Архитектура и программная модель CUDA. М. : Изд-во Московского университета, 2012. С. 30.
3. В. М. Вержбицкий Основы численных методов. — М.: Высшая школа, 2009. — 840 с. — ISBN 9785060061239

4. В. И Крылов Вычислительные методы высшей математики: в т. 1 / В. И. Крылов, В. В. Бобков, П. И. Монастырный. Минск: Издательство «Высшая школа», 1972. 219 с.
5. Дж.Х. Уилкинсон Справочник алгоритмов на языке АЛГОЛ. Линейная алгебра. Перевод с английского под ред. д-ра техн. наук проф. Ю. И. Топчеева. М., «Машиностроение», 1976 г. С. 216–221.