УДК 519.684.4

ИССЛЕДОВАНИЕ ПАРАЛЛЕЛЬНЫХ АЛГОРИТМОВ ПРОГОНКИ И ЦИКЛИЧЕСКОЙ РЕДУКЦИИ НА ГРАФИЧЕСКОМ ПРОЦЕССОРЕ С ИСПОЛЬЗОВАНИЕМ РАЗЛИЧНЫХ ТИПОВ ПАМЯТИ И ПОДХОДОВ К ХРАНЕНИЮ ДАННЫХ

© Погорельских К.С., Логанова Л.В.

e-mail: sekih@yandex.ru

Самарский национальный исследовательский университет имени академика С.П. Королёва, г. Самара, Российская Федерация

Для решения трехдиагональных СЛАУ традиционно используют метод прогонки и метод циклической редукции. В основе этих методов лежат простые арифметические операции, что позволяет реализовывать их на графическом процессоре. Метод прогонки характеризуется низкой вычислительной сложностью, но плохо распарарливается для решения одной системы.Преимуществом метода циклической редукции является его высокая степень параллелизма [1].

Существенное время выполнения программы на графическом процессоре занимает работа с памятью [2]. Глобальная память видеокарты работает медленно, что обращает внимание исследователей на возможность использования разделяемой памяти. Она значительно быстрее глобальной, но её объем невелик, поэтому данные в эту память необходимо загружать порционно.

В данной работе исследуются реализации алгоритма прогонки, а также последовательной и параллельной версий алгоритма циклической редукции с использованием различных комбинаций глобальной и разделяемой памяти GPU. Распараллеливание методов проводится с использованием программно-аппаратной архитектуры CUDA. Экспериментальные исследование проводились на суперкомпьютере «Сергей Королев» при количестве систем 5000 до 40000 размерностью 4095.

Алгоритм прогонки был реализован на CPU, на GPUс использованием только глобальной памяти видеокарты и на GPU с порционной загрузкой всех данных в разделяемую память. Был посчитано ускорение относительно последовательной версии на CPU. Среднее ускорение версии на GPU с использованием только глобальной памяти равно 1,26. Среднее ускорение с использованием разделяемой памяти – 1,86.

На основе версии алгоритма с использованием разделяемой памяти был разработан алгоритм с подходом к хранению коэффициентов в глобальной памяти упорядоченно по индексам. Суть подхода в том, чтобы хранить коэффициенты не по матрицам (сначала все коэффициенты первой матрицы, затем второй, третьей и т.д.), а по номерам (сначала все нулевые элементы всех матриц, затем первые, второе и т.д.). Этот алгоритм показал среднее ускорение 5,05 относительно последовательной версии на CPU. Причем с увеличением количества матриц ускорение возрастало.

Алгоритм последовательной циклической редукции SERICR [3] был реализован на CPU, на GPU с использованием только глобальной памяти видеокарты и на GPU с порционной загрузкой всех данных в разделяемую память. Было посчитано ускорение относительно последовательной версии на CPU. Среднее ускорение версии на GPU с использованием только глобальной памяти равно 0,18. Среднее ускорение с использованием разделяемой памяти – 0,27. Такой результат объясняется низким параллелизмом метода и необходимостью многократного перерасчета коэффициентов.

Вычисления с использованием алгоритма SERICR на графическом процессоре не оправданы.

Алгоритм параллельной циклической редукцииPARACRбыл реализован на GPU с использованием только глобальной памяти видеокарты, на GPU с порционной загрузкой всех данных в разделяемую память и на GPUc порционной загрузкой только части коэффициентов в разделяемую память. Было посчитано ускорение относительно реализации последовательного алгоритма циклической редукции SERICR на CPU. Эксперимент показал ускорение 1,97 версии с использованием только глобальной памяти, ускорение 1,20 с полной загрузкой в разделяемой память и 2,01 с частичной загрузкой в разделяемую память. Результат объясняется тем, что многие данные в алгоритме циклической редукции используются только один раз. При загрузке в разделяемую память таких данных тратится дополнительное время, которое ничем не компенсируется. Лучший результат показала версия с загрузкой в разделяемую память только тех коэффициентов, которые используются многократно.

На основе лучшей версии алгоритма PARACR по аналогии с алгоритмом прогонки был реализован подход к хранению коэффициентов в глобальной памяти упорядоченно по индексам. Реализация показала среднее ускорение 1,81 относительно последовательного алгоритма циклической редукции SERICR на CPU. Ускорение ниже, чем у исходной версии. Это объясняется тем, что в алгоритме PARACR при подсчете новых коэффициентов происходит многократное обращение к соседним коэффициентам в глобальной памяти. А подход к хранению коэффициентов в глобальной памяти упорядоченно по индексам приводит к тому, что соседние коэффициенты располагаются в памяти далеко друг от друга.

При сравнении лучших реализаций алгоритмов прогонки и циклической редукции на совокупности систем лучший результат показывает алгоритм прогонки. Он использует в 15 раз меньше времени на вычисления, чем алгоритм PARACR. При решении одной системы большой размерности лучший результат показывает алгоритм циклической редукции. Он требует в среднем в 7 раз меньше времени на вычисления, чем алгоритм прогонки, что объясняется его высоким параллелизмом [4].

Таким образом, для решения совокупности систем оправдано использовать реализацию алгоритма прогонки с использованием разделяемой памяти и хранением коэффициентов упорядоченно по индексам. Для решения одной системы лучше всего подходит параллельный алгоритм циклической редукции с частичной загрузкой в разделяемую память и хранением коэффициентов упорядоченно по системам.

Представляет интерес разработка интегрированного алгоритма на основе методов прогонки и циклической редукции, объединяющего в себе достоинства обоих алгоритмов.

Библиографический список

1. Шинкарук, Д.Н. Анализ эффективности применения технологии CUDA для решения систем линейных уравнений с трехдиагональными матрицами в задачах расчета цен опционов [Текст] / Д.Н. Шинкарук, Ю.А. Шполянский, М.С. Косяков // Изв. вузов. Приборостроение. – 2012. – №10. – 6 с.

2. Технология CUDA в примерах: введение в программирование графических процессоров / пер. с анг. А.А. Слинкина, науч. ред. А.В. Боресков. – М.: ДМК Пресс, 2011. – 232 с.

3. Хокни, Р. Параллельные ЭВМ. Архитектура, программирование и алгоритмы / Р. Хокни, К. Джессхоуп; пер. с анг. Д.И. Абашкина. – М.: Радио и связь, 1986. – 392 с.

4. Ярмушкин, С.В. Исследование параллельных алгоритмов решения трехдиагональных систем линейных алгебраический уравнений [Текст] / С.В. Ярмушкин, Д.Л. Головашкин // Вестник Самарского Государственного Технического Университета сер. Физико-математические науки. – 2004. – №26. – С. 5.