ПРОЕКТ ПРАКТИКУМА ПО КУРСУ ВЫСОКОПРОИЗВОДИТЕЛЬНЫХ ВЫЧИСЛЕНИЙ, ОСНОВАННЫЙ НА CUDA

А.О. Курсиков

Тульский госуниверситет

Введение

В настоящее время вычисления на GPU приобретают все большую актуальность. В связи с этим необходимо введение учебных курсов для подготовки студентов по направлению “Прикладная математика и информатика” для обучения основам программирования для GPU. Уже издана учебная литература [1], но ввиду недостаточного количества простых примеров, приведенных в данной книге, она не может быть использована в качестве основного учебного пособия для студентов третьего курса. В рамках создания курса “Вычислительные системы и параллельная обработка данных” в Тульском государственном университете на механико-математическом факультете разрабатывается учебный практикум, который должен охватывать все основные аспекты программирования вычислительных задач на современных параллельных архитектурах GPU. На практических занятиях будет предложено реализовать алгоритмы, изложенные в курсе “Численные методы” [2]. Такой выбор дает ряд преимуществ: у слушателей курса не будет необходимости изучать новые алгоритмы и будет возможность сравнить GPU-реализации с последовательными реализациями на CPU. В учебную программу планируется включить следующие задачи:

1) базовые задачи линейной алгебры;
2) численное интегрирование;
3) решение СЛАУ методом простой итерации;
4) более сложные задачи.

Простейшие задачи линейной алгебры

Для введения в параллельное программирование будет предложен классический пример сложения больших векторов. Одной из целей этого примера является знакомство со средой выполнения. Следующей лабораторной работой будет реализация параллельного суммирования (редукции), которая в дальнейшем будет использована в задаче численного интегрирования. Ввиду исчерпывающего описания в литературе, возможно предоставление для самостоятельной работы задачи перемножения матриц.

Численное интегрирование

Для численного интегрирования предлагается использовать простейшие формулы прямоугольников, трапеций, Симпсона и более сложные формулы Ньютона–Котеса. Например, для метода средних прямоугольников имеем следующую расчетную формулу:

$$\int_a^b f(x)dx \approx \sum_{i=1}^{n} hf\left(x_i + \frac{h}{2}\right),$$

где f – подынтегральная функция, x_i – середина i-го интервала длины h.

Выполнены следующие реализации:

1) с параллельным вычислением значений подынтегральной функции на участках разбиения отрезка интегрирования и последующим последовательным суммированием результатов на CPU;

2) с параллельным суммированием (с применением библиотеки CUDPP, собственных реализаций с использованием разделяемой памяти (shared memory) и устранением дивергентного ветвления (divergent branching));

3) с использованием формул Ньютона–Котеса более высоких степеней.

Они позволяют последовательно подойти к изучению основ программирования на GPU. При этом в силу того, что рассматриваемые алгоритмы задачи численного интегрирования просты и изучены в курсе численных методов, трудностей, связанных с их пониманием, возникать не должно.

Решение СЛАУ методом простой итерации

Согласно методу простой итерации исходная система $Ax = b$ приводится к виду

$$x = Bx + \beta,$$

где x – вектор переменных, B – приведенная матрица системы, β – приведенный вектор свободных коэффициентов. Затем задается начальное значение x_0 и находится следующее приближение $x_1 = Bx_0 + \beta$. Процесс продолжается до достижения требуемой точности. Критерием остановки может служить близость по норме двух последних приближений.

Для сходимости процесса необходимо и достаточно того, чтобы все собственные значения матрицы B по модулю были меньше единицы

Пример решения СЛАУ методом простой итерации тоже крайне прост, и на нем можно продемонстрировать все основные аспекты использования ресурсов GPU. Были сделаны четыре реализации этого алгоритма:

1) “наивная” реализация, в которой параллельно выполняется операция скалярного произведения векторов;

2) разбиение задачи на блоки для уменьшения числа обращений к глобальной памяти, транспонирование матрицы для непрерывного чтения (coalesced reading);

3) использование разделяемой памяти;

4) выполнение базовых операций линейной алгебры с использованием библиотеки CUBLAS.

Как и в предыдущем случае, пример усложняется от реализации к реализации и освещает основные аспекты программирования на GPU. Особый интерес представляет третья реализация, поскольку при блочном решении задачи необходимо подобрать оптимальные параметры блока (количество используемых переменных и строк исходной матрицы). В противном случае получаем потерю производительности. Этот пример наглядно демонстрирует зависимость производительности вычислений от архитектуры GPU.

Более сложные задачи

В разрабатываемом учебном курсе центральное место занимают упомянутые выше задачи, поскольку они охватывают базовые основы программирования для GPU. Рассматривается вопрос о введении в курс более сложных задач, например, решение СЛАУ с трехдиагональной матрицей или реализацию одной из базовых операций с разреженными матрицами.

Литература

1. Боресков А.В. и др. Параллельные вычисления на GPU. Архитектура и модель CUDA: Учебное пособие. М.: Издательство МГУ, 2012. ISBN 978-5-211-06340-2.
2. Калиткин Н.Н. Численные методы. СПб.: БХВ-Петербург, 2011. ISBN 978-5-775-0500-0.