ИНФОРМАТИКА

УДК 004.657

ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ. ТЕХНОЛОГИЯ CUDA

Асп. ***Кудухов А. Н.***
Северо-Кавказский горно-металлургический институт
(государственный технологический университет)
г. Владикавказ, РСО-Алания, Россия

Разбор и ознакомление с архитектурой параллельного вычисления CUDA. CUDA – это архитектура параллельных вычислений от NVIDIA, позволяющая существенно увеличить вычислительную производительность благодаря использованию GPU.

Внутри каждой большой задачи сидит маленькая, пытающаяся пробиться наружу.

Из закона Мерфи

Нынешнее время – время обработки больших объемов данных, требующих больших вычислительных мощностей.

Со времен появления понятия распараллеливание, разработчики стараются писать код таким образом, чтобы отдельные его части выполнялись параллельно, а не последовательно, но без потери связи между ними. Но, в реалиях нашего времени зачастую вычислительных мощностей процессора не хватает, так как рост частот универсальных процессоров упёрся в физические ограничения и высокое энергопотребление, и увеличение их производительности всё чаще происходит за счёт размещения нескольких ядер в одном чипе, что влечет за собой более высокое энергопотребление и более сложную архитектуру [1].

Теперь обратимся к видеокартам. У них своя собственная архитектура вычислений, и до недавнего времени было довольно трудно представить, чтобы видеокарты помогали процессору в каких-либо вычислениях, кроме как для них предназначающихся (обработка графики, поддержка GUI, векторные вычисления). Но теперь появились технологии, позволяющие производить вычисления с использованием графических процессоров, поддержи-

вающих технологию GPGPU (произвольных вычислений на видеокартах). Одна из таких технологий – CUDA.

CUDA – это архитектура параллельных вычислений от NVIDIA, позволяющая существенно увеличить вычислительную производительность благодаря использованию GPU(блок графического процессора). Суть этой технологии заключается в том, что если код будет написан таким образом, что он будет хорошо распараллеливаться, то вычисления будут протекать с гораздо большей скоростью, чем на обычном процессоре [2].

Трудоемкие алгоритмы, обработка больших массивов данных, вычисления, связанные с математическими моделями и формулами, теперь могут выполняться в десятки, если не в сотни раз быстрее, и при этом не обязательно покупать огромные вычислительные центры и мощные сервера.

Архитектура CUDA

Одной из важнейших характеристик любого вычислительного устройства является его быстродействие. Для математических расчетов быстродействие обычно измеряется в количестве floating-point операций в секунду. При этом довольно часто рассматривается так называемое пиковое быстродействие, т. е. максимальное возможное число операций с вещественными величинами в секунду.

Для персонального компьютера быстродействия обычно напрямую связано с тактовой частотой центрального процессора. Процессоры архитектуры x86 за время с момента своего появления в июне 1978 года увеличили свою тактовую частоту в 700 раз.

Год	Тактовая частота	Процессор
1978	4,77 MHz	Intel 8086
2004	3.46 GHz	Intel Pentium 4
2005	3.8 GHz	Intel Pentium 4
2006	2.333 GHz	Intel Core Duo T2700
2007	3 GHz	Intel Core 2 Duo E6700
2008	3.33 GHz	Intel Core 2 Duo E8600

Однако если внимательно посмотреть на динамику роста частоты CPU, то становится заметно, что в последние годы рост частоты заметно замедлился, но зато появляется новая тенденция – создание многоядерных процессоров и систем и увеличение числа ядер в процессоре.

Максимальное ускорение, которое можно получить от распараллеливания программы на N процессоров, дается законом Амдала [1]:

$$S = \frac{1}{(P) + \frac{1-P}{N}},$$

где N – количество процессоров,

P – объем вычислений, который может быть получен последовательными вычислениями,

$1–P$ – объем вычислений, который может быть распараллелена идеально.

Таким образом, если мы можем распараллелить ¾ всей программы, то максимальный выигрыш составит 4 раза.

Используемая для распараллеливания технология CUDA предлагает потоковую модель вычислений. Вычислительная архитектура CUDA основана на концепции «одна команда, множество данных» (*Single Instruction Multiple Data, SIMD*).

Концепция SIMD подразумевает, что одна инструкция позволяет одновременно обработать множество данных. Мультипроцессор – это многоядерный SIMD-процессор, позволяющий в каждый определенный момент времени выполнять на всех ядрах только одну инструкцию.

Основная цель архитектуры – это размещение на кристалле несколько десятков процессорных ядер с собственной памятью, каждая из которых выполняет несколько сотен программных потоков. Процессорные ядра, называемые в CUDA – терминологии мультипроцессорами, имеют собственный доступ к глобальной памяти, расположенной на видеоплате и устройство обменивается данными с CPU через шину PCI-Express.

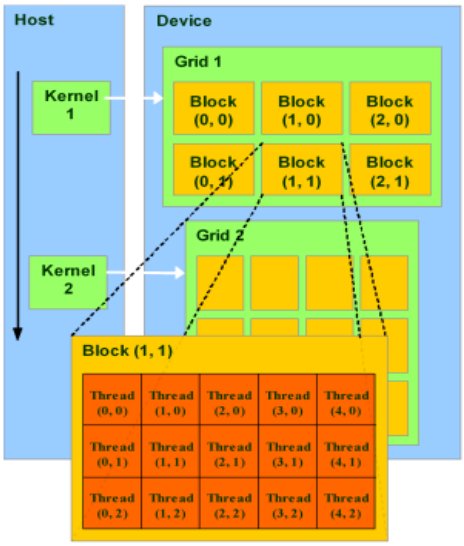
Есть два факта, их можно назвать законом вычислительной техники, который составляют теоретическую основу востребованности высокопараллельных вычислительных архитектур. Потребляемая мощность процессора пропорциональна примерно квадрату тактовой частоты, примерно степени 2,5. То есть, процессор с тактовой частотой 3 ГГц потребляет больше, чем 9 процессоров частотой 1 ГГц. Таким образом, для параллельной программы энергоэффективность массы мелких процессоров будет выше в три раза. Иными словами, многоядерный процессор, с той же потребляемой энергией, будет в три раза производительнее при исполнении параллельного кода.

Второй факт относится уже к программному обеспечению и алгоритмам. Если задача распараллеливается, то есть, допускает реализацию с помощью параллельного алгоритма, который может исполняться одновременно на нескольких процессорах, то чем на большее количество потоков она уже распараллелена, то тем больше вероятность, что задача может быть распараллелена на ещё большее количество потоков. Например, большое количество задач не допускает параллельной реализации, многие распараллеливаются только на два потока, а если алгоритм допускает три потока, то весьма вероятно, что он распараллелится и на четыре и на пять и на шесть потоков.

Такая ситуация типична для вычислительных задач с большими объемами данных, которые по своей природе, в принципе, допускают параллельный алгоритм решения. А если такого нет, то можно и не надеяться на увеличение производительности в обозримом будущем, ибо тактовая частота современных CPU растет очень медленно. В таких случаях, выгодно реализовать многопоточный алгоритм, пусть он будет и на порядок более вычислительно затратным, все равно, площади кристаллов девать некуда и, на мультипроцессорной системе такой, сам по себе малоэффективный алгоритм, будет работать быстрее. Но, уже имея задачу с тысячами потоками, можно заранее адаптировать вычислительную архитектуру с учетом мультипоточности, сконцентрироваться на общей производительности программы, а не каждой отдельной нити, сильно сэкономив на многих традиционных архитектурных деталях.

Рассмотрим вычислительную модель CUDA

Модель программирования в CUDA предполагает группирование потоков. Потоки объединяются в блоки потоков (thread block) – одномерные или двумерные сетки потоков, взаимодействующих между собой при помощи разделяемой памяти и точек синхронизации. Программа (ядро, kernel) исполняется над сеткой (grid) блоков потоков (thread blocks). Одновременно исполняется одна сетка. Каждый блок может быть одно-, двух- или трехмерным по форме, и может состоять из 512 потоков на текущем аппаратном обеспечении [3].



Вычислительное устройство GPU.

Блоки потоков выполняются в виде небольших групп, называемых варп (warp), размер которых – 32 потока. Это минимальный объём данных, которые могут обрабатываться в мультипроцессорах. И так как это не всегда удобно, CUDA позволяет работать и с блоками, содержащими от 64 до 512 потоков.

Группировка блоков в сетки позволяет уйти от ограничений и применить ядро к большему числу потоков за один вызов. Это помогает и при масштабировании. Если у GPU недостаточно ресурсов, он будет выполнять блоки последовательно. В обратном случае, блоки могут выполняться параллельно, что важно

для оптимального распределения работы на видеочипах разного уровня, начиная от мобильных и интегрированных.

Вывод

На сегодняшний день компания NVIDIA – это не только графические чипы в традиционном их определении. Технология CUDA уже превратилась в индустриальный стандарт для параллельных вычислений. CUDA даёт разработчику возможность по своему усмотрению организовывать доступ к набору инструкций графического ускорителя и управлять его памятью. Графический ускоритель с поддержкой CUDA становится мощной программируемой открытой архитектурой, приближаясь к сегодняшним центральным процессорам.

ЛИТЕРАТУРА

1. *Боресков А.В., Харламов А.А.* Основы работы с технологией CUDA. М.: ДМК – Пресс, 2010. 232 с.

2. http://nvworld.ru/articles/cuda_parallel/page3/#h10c0669665bfddbf68bba2171fee6d13 – Параллельные вычислительные процессоры NVIDIA: настоящее и будущее.

3. http://www.lki.ru/text.php?id=5942 – Куда ведет CUDA: практическое применение.

4. http://www.nvidia.ru/object/cuda_state_university_courses_new_ru.html – Курс лекций по CUDA.

5. http://www.nvidia.ru/object/what_is_cuda_new_ru.htm Что такое CUDA.