УДК 004.021

ИЗУЧЕНИЕ ТЕХНОЛОГИИ NVIDIA CUDA НА ПРИМЕРЕ АЛГОРИТМА ХЕШИРОВАНИЯ MD5

Гребенюк Е.О., Ишмухаметов Ш.Т.

ФГАОУ ВПО Казанский (Приволжский) федеральный университет,
420008, г. Казань, ул. Кремлевская, д.18.

e-mail: llceceron@mail.ru, Shamil.Ishmukhametov@kpfu.ru

поступила в редакцию 28 августа 2013 года

Аннотация

В настоящее время развивается технологии, связанные с вычислением на графических видеокартах с использованием распараллеливания. В связи с этим компания NVIDIA разработала инструменты позволяющие использовать возможности графических процессоров для простейших вычислений, что при распараллеливании существующих программ позволяет существенно уменьшить время выполнения.

В работе разрабатывался программный комплекс для реализации алгоритма хеширования MD5 с использованием технологии NVIDIA CUDA. Данный комплекс позволяет выполнить реализацию алгоритма с распараллеливанием на большое количество потоков, что ускоряет выполнение алгоритма.

Ключевые слова: *CUDA, NVIDIA CUDA, MD5, хеширование, алгоритм MD5.*

Введение. CUDA (англ. Compute Unified Device Architecture) – программно-аппаратная архитектура параллельных вычислений, которая позволяет существенно увеличить вычислительную производительность благодаря использованию графических процессоров фирмы NVIDIA.

CUDA SDK позволяет программистам реализовывать на специальном упрощённом диалекте языка программирования Си алгоритмы, выполнимые на графических процессорах NVIDIA, и включать специальные функции в текст программы на Си. Архитектура CUDA даёт разработчику возможность по своему усмотрению организовывать доступ к набору инструкций графического ускорителя и управлять его памятью [1].

Основная часть. Вычисления на GPU или GPGPU заключаются в использовании GPU (графического процессора) для универсальных вычислений в области науки и проектирования. GPU вычисления представлены совместным использованием CPU и GPU в гетерогенной модели вычислений. Стандартная часть приложения выполняется на CPU, а более требовательная к вычислениям часть обрабатывается с GPU ускорением. С точки зрения пользователя приложение работает быстрее, потому что оно использует высокую производительность GPU для повышения производительности [2].

Составлена программа выполнения алгоритма хеширования MD5 на CPU. Далее исследованы материалы по программированию с использованием CUDA runtime API и реализована аналогичная программа, но уже используя возможности GPU.

Платформа параллельных вычислений CUDA обеспечивает набор абстракций, позволяющих выражать как параллелизм данных, так и параллелизм задач на уровне мелких и крупных структурных единиц. Программист может выбрать средства разработки: языки высокого уровня, такие как C, C++, Fortran или же драйверы API, такие как DirectX™-11 Compute [3].

Алгоритм MD5. Данный алгоритм принимает на входе сообщение произвольного размера и выдает в результате 128-битовый «отпечаток» («fingerprint») или цифровую подпись («message digest»). Предполагается, что потребуются нереальный объем вычислений для создания двух сообщений, цифровые подписи которых совпадут, или подбора сообщения по имеющейся цифровой подписи. Алгоритм MD5 предназначен для создания цифровой

подписи (сигнатуры), когда требуется безопасно «сжать» большой файл перед тем, как шифровать его с использованием закрытого (секретного) ключа в системах с открытыми ключами типа RSA.

Алгоритм MD5 разработан для обеспечения достаточной скорости на 32-битовых машинах. В дополнение к этому MD5 не требует использования больших таблиц подстановки; кодирование алгоритма может быть достаточно компактным.

MD5 является расширением алгоритма цифровых подписей MD4 [1,2]. MD5 несколько медленней, чем MD4, но устроен более «консервативно». Причиной разработки MD5 послужило ощущение того, что алгоритм MD4 может быть адаптирован для применения быстрее, нежели предполагалось; поскольку при разработке MD4 основным требованием была высокая скорость, он находится «на самом краю» допустимости в плане устойчивости к криптоаналитическим атакам. MD5 немного проигрывает в скорости, но существенно выигрывает в безопасности. Алгоритм MD5 открыт для обозрения (public domain).

Алгоритм цифровых подписей MD5 прост в реализации и создает «отпечатки» (fingerprint) или цифровые подписи для сообщений произвольной длины. Предполагается, что для создания двух сообщений с одинаковыми сигнатурами потребуется порядка 2^64 операций, а для подбора сообщения по имеющейся сигнатуре – порядка 2^128 операций [4].

Заключение. При реализации возникли трудности с программой, которая использовала вычисления на GPU, прежде всего это было связано с особенностями данной технологии (необходимость выделения памяти, загрузка и изъятие из памяти данных на GPU).

Для тестирования был взят набор различных исходных данных (Таблица 1).

Графики зависимостей:

1) Время выполнения программы на CPU в зависимости от количества символов передаваемых в качестве начальных данных:

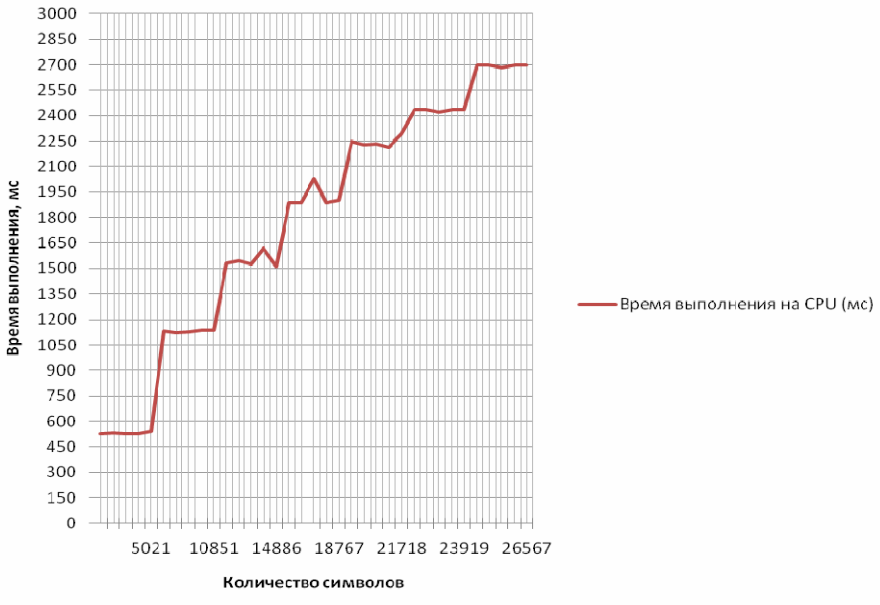

Рисунок 1. – Время выполнения программы на CPU.

Из графика видно, что при увеличении количества символов в сообщении время выполнения растет. Что говорит об ограниченном количестве информации обрабатываемой за такт на CPU.

Таблица 1. – Результаты выполнения программ.

Время выполнения на CPU (мс)	Время выполнения на GPU (мс)	Длина сообщения (символов)
2699	1382	26567
2699	1385	
2683	1391	
2698	1380	
2698	1400	
2434	1234	23919
2433	1220	
2418	1235	
2433	1229	
2434	1237	
2293	1008	21718
2215	1007	
2231	1003	
2230	1010	
2246	985	
1903	839	18767
1887	837	
2028	840	
1888	835	
1887	838	
1513	630	14886
1618	634	
1526	627	
1547	633	
1531	630	
1134	231	10851
1135	228	
1125	235	
1120	234	
1129	232	
543	113	5021
528	115	
530	114	
534	110	
528	112	

2) Время выполнения программы с использованием GPU в зависимости от количества символов передаваемых в качестве начальных данных:

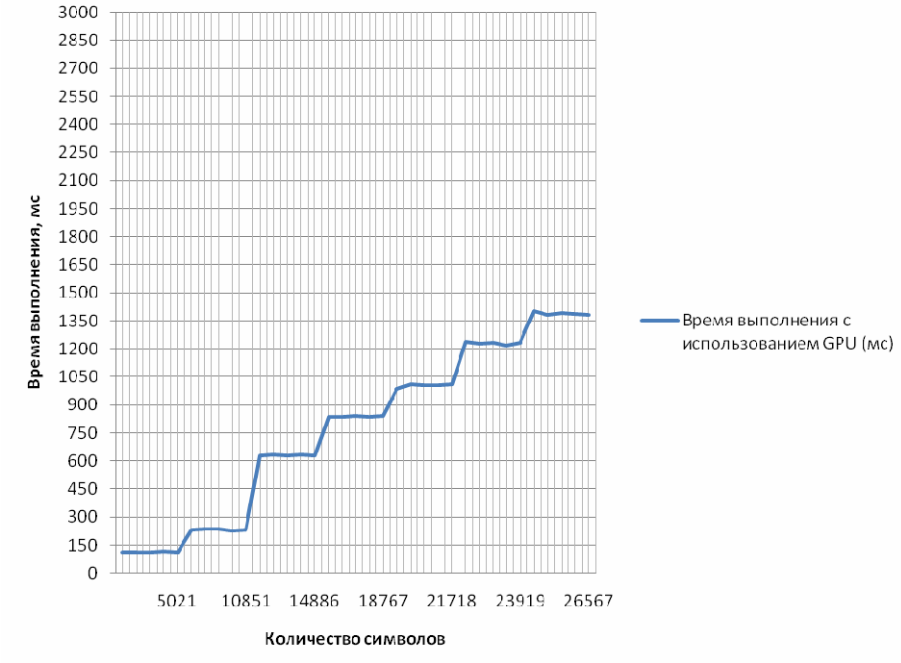

Рисунок 2. – Время выполнения программы с использованием GPU.

Из графика видно, что так же время выполнения растет, как и на предыдущей зависимости. Но следует заметить, что этот рост происходит значительно медленнее, т.е. времени выполнения требуется меньше на ту же самую последовательность исходных данных.

3) Сравнение времени выполнения программы на CPU и с использованием GPU в зависимости от количества символов в исходных данных:

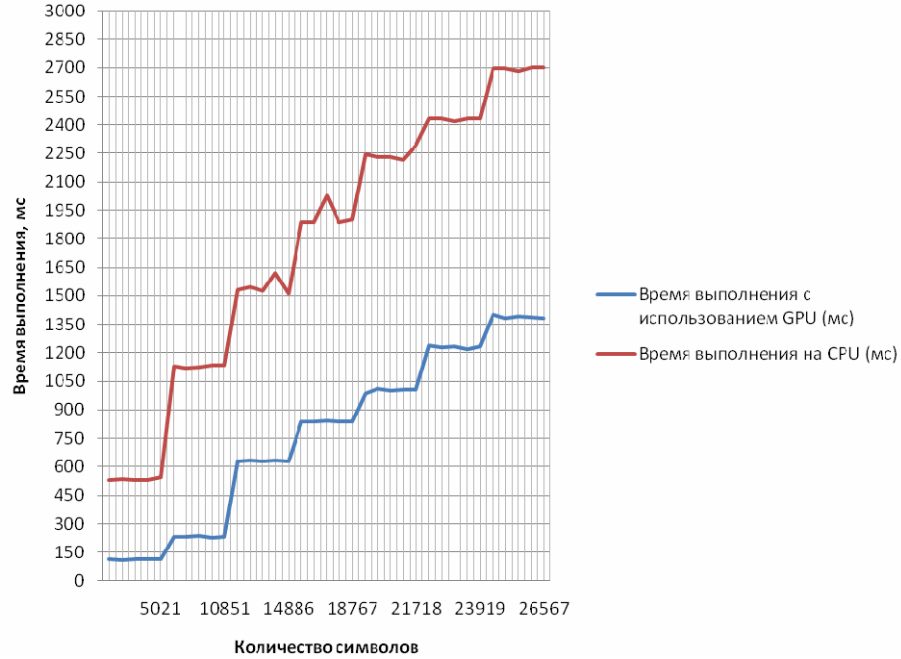

Рисунок 3. – Сравнение времени выполнения программы на CPU и с использованием GPU.

Из данного графика видно, что программа, использующая для вычислений GPU, выполняется быстрее при выполнении с одинаковым набором исходных данных.

Из полученных результатов можно сделать вывод, что если есть возможность распараллеливания алгоритма выполнения поставленной задачи, то её можно попытаться

реализовать с использованием GPU. С ожидаемым при этом приростом производительности новой программы, однако, она не будет выше в столько раз, во сколько количество процессоров на GPU превосходит аналогичное количество процессоров CPU (Закон Амдаля) [5].

Согласно этому закону, ускорение выполнения программы за счёт распараллеливания её инструкций на множестве вычислителей ограничено временем, необходимым для выполнения её последовательных инструкций.

Так же следует заметить, что одна и та же программа может выполняться быстрее, если использовать более новое аппаратное обеспечение, на данный момент средний прирост, достигнутый при разработке программ с использованием GPU около 10 раз по сравнению с аналогичной программой на CPU.

Небольшим недостатком данной технологии является то, что программы могут выполняться только на видеокартах от NVIDIA. Но это легко компенсируется доступностью данных комплектующих и даже из самой низкой ценовой категории они позволяют увеличить скорость выполнения программы.

Список литературы

1) Интернет-ресурс: CUDA. Материал из Википедии – свободной энциклопедии. http://ru.wikipedia.org/wiki/CUDA (Дата обращения 28.09.2013).
2) Интернет-ресурс: Центр параллельных вычислений. http://parallelcompute.sourceforge.net/index_ru.php (Дата обращения 28.09.2013).
3) Интернет-ресурс: Параллельные вычисления CUDA. http://www.nvidia.ru/object/cuda-parallel-computing-ru.html (Дата обращения 28.09.2013).
4) Интернет-ресурс: RFC 1321 – Алгоритм цифровых подписей MD5. http://rfc2.ru/1321.rfc (Дата обращения 28.09.2013).
5) Интернет-ресурс: Закон Амдала. Материал из Википедии – свободной энциклопедии. http://ru.wikipedia.org/wiki/%C7%E0%EA%EE%ED_%C0%EC%E4%E0%EB%E0 (Дата обращения 28.09.2013).