**КОМПЬЮТЕРНЫЕ ИССЛЕДОВАНИЯ
И МОДЕЛИРОВАНИЕ 2010 Т. 2 № 3 С. 323–326**

Ки&М

МАТЕМАТИЧЕСКИЕ ОСНОВЫ И ЧИСЛЕННЫЕ МЕТОДЫ МОДЕЛИРОВАНИЯ

УДК: 004

Реализация клеточных автоматов «игра “Жизнь”» с применением технологий CUDA и OpenCL

А. Е. Алексеенко[a], А. М. Казённов[b]

Московский физико-технический институт,
Россия, 141700, Долгопрудный, пер. Институтский, 9

E-mail: [a] al42and@gmail.com, [b] kazennov@gmail.com

Получено 9 сентября 2010 г.

В данной статье проанализирован опыт преподавания курса «Программирование на CUDA и OpenCL» для участников ежегодной межвузовской молодежной школы по высокопроизводительным вычислениям МФТИ-2010. В статье разобраны как содержимое лекций и семинарские задачи, так и особенности преподнесения материала. Обсуждаются результаты, полученные учащимися при выполнении практических задач. Приводится сравнение быстродействия различных алгоритмов реализации клеточных автоматов «игра “Жизнь”» с применением технологий CUDA и OpenCL.

Ключевые слова: CUDA, OpenCL, GPGPU, клеточные автоматы, летняя школа, практикум

CUDA and OpenCL implementations of Conway’s Game of Life cellular automata

A. E. Alekseenko, A. M. Kazennov

Moscow Institute of Physics and Technology, 9 Institutskii per, Dolgoprudny, 141700, Russia

Abstract. — In this article the experience of reading “CUDA and OpenCL programming” course during high perfomance computing summer school MIPT-2010 is analyzed. Content of lectures and practical tasks, as well as manner of presenting of the material are regarded. Performance issues of different algorithms implemented by students at practical training session are dicussed.

Keywords: CUDA, OpenCL, GPGPU, cellular automata, summer school, workshop

Citation: *Computer Research and Modeling*, 2010, vol. 2, no. 3, pp. 323–326 (Russian).

Летняя школа по высокопроизводительным вычислениям МФТИ 2010 года предназначалась для освоения студентами основных технологий для распараллеливания на системах как с общей, так и с разделяемой памятью. В данной статье рассказывается опыт проведения курсов по технологии CUDA и стандарту OpenCL.

Применяя опыт предыдущих Школ, было принято решение о проведении однодневных блоков по каждому из курсов. В блок входило 6 академических часов лекций и 4 часа семинаров. Для семинаров были подготовлены примеры базовых программ, таких как вычисление числа «Пи» методом интегрирования четверти круга и задача перемножения матриц. Первая предназначалась для иллюстрации минимального набора действий, необходимых для использования графических адаптеров для не графических расчетов. Вторая предназначена для обучения правильному разбиению задачи на вычислительные группы (блоки) и разбиению блоков на отдельные вычислительные потоки (треды), а также на правильное индексирование элементов вычислений.

Принципиальной является также последовательность чтения курсов. Первым был прочитан курс по CUDA, так как в нём содержится информация об особенностях аппаратной архитектуры графических адаптеров, основных ограничениях и методах оптимизации. После курса по CUDA в курсе по OpenCL остаётся рассказать только об особенностях инициализации и компиляции вычислительных ядер. Таким образом, оба курса гармонично дополняют друг друга, полностью покрывая возможности использования систем с общей памятью (многоядерные процессоры, графические адаптеры, многопроцессорные системы).

После завершения всех курсов участникам школы было предложено самостоятельно написать программу, реализующую клеточный автомат «игра “Жизнь”» и распараллелить его, используя любую из пройденных технологий. Основной проблемой при решении данной задачи является разбиение задачи на блоки. Необходимо разбить на блоки так, чтобы алгоритм не зависел от размера «поля» и при этом был быстрым и эффективным. Приведём два примера возможного разбиения.

Первый — разбить трёхмерные кубики так, чтобы размер не превышал $8 \times 8 \times 8$. Однако такое разбиение обладает существенными недостатками.

1. Оно плохо адаптируется к размеру мира.
2. Оно не позволяет в полной мере использовать преимущества блочного чтения из памяти.
3. Обладает достаточно сложной индексацией.

К преимуществам данного разбиения можно отнести то, что оно обеспечивает достаточно большой размер блока.

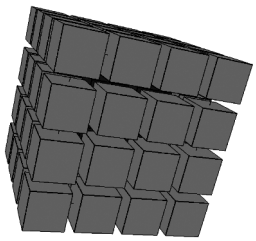

Рис. 1. Первая схема разбиения, считываемая из памяти область

Второй способ — задача разбивается на отдельные столбцы толщиной в одну клетку и длиной в одно из измерений «поля» (предлагается брать среднее при условии, что оно не превышает 512). Такое разбиение лишено всех недостатков первого, но не содержит и его преимуществ.

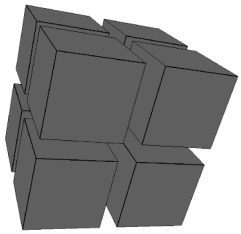

Рис. 2. Первая схема разбиения, вычисляемая область

Например, если «поле» имеет размерность $8 \times 8 \times 1024$, то первое разбиение будет иметь блок $8 \times 8 \times 8 = 512$ тредов, что позволит эффективно использовать видеокарту. Второе будет иметь блок размером в 8 тредов, и вычислительные ресурсы видеокарты будут использованы далеко не полностью. С другой стороны, задача с размером мира $100 \times 100 \times 100$ будет отлично решаться на втором разбиении и вызовет трудности на первом.

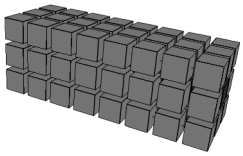

Рис. 3. Вторая схема разбиения, считываемая из памяти область

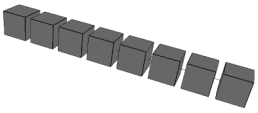

Рис. 4. Вторая схема разбиения, вычисляемая область

Таким образом, разбиение необходимо выбирать исходя из специфики задачи. В решении жюри использовалась вторая схема, так как чаще всего «поле» в подобных клеточных автоматах – кубическое или достаточно близко к таковому. На рис. 5 приводится график зависимости времени расчета от размера ребра куба для технологий OpenCL и CUDA.

Одним из участников Школы (Балычевым Андреем из РГУ им. Канта) было написано решение, на 15 % превосходившее по скорости решение жюри. Он использовал модификацию второго метода разбиения, но вместо столбца $1 \times 1 \times Z$ брал столбец $2 \times 2 \times Z$, что позволило уменьшить отношение площади поверхности одного блока к его объёму, т. е. отношение количества обращений к памяти к собственно вычислениям. Данный метод можно применять и далее (при размере «поля» по одной из координат 512 можно брать блоки $9 \times 9 \times 512$, что ещё сильнее уменьшит это соотношение).

Всего по итогам школы проект на CUDA был реализован 7 участниками, на OpenCL — двумя.

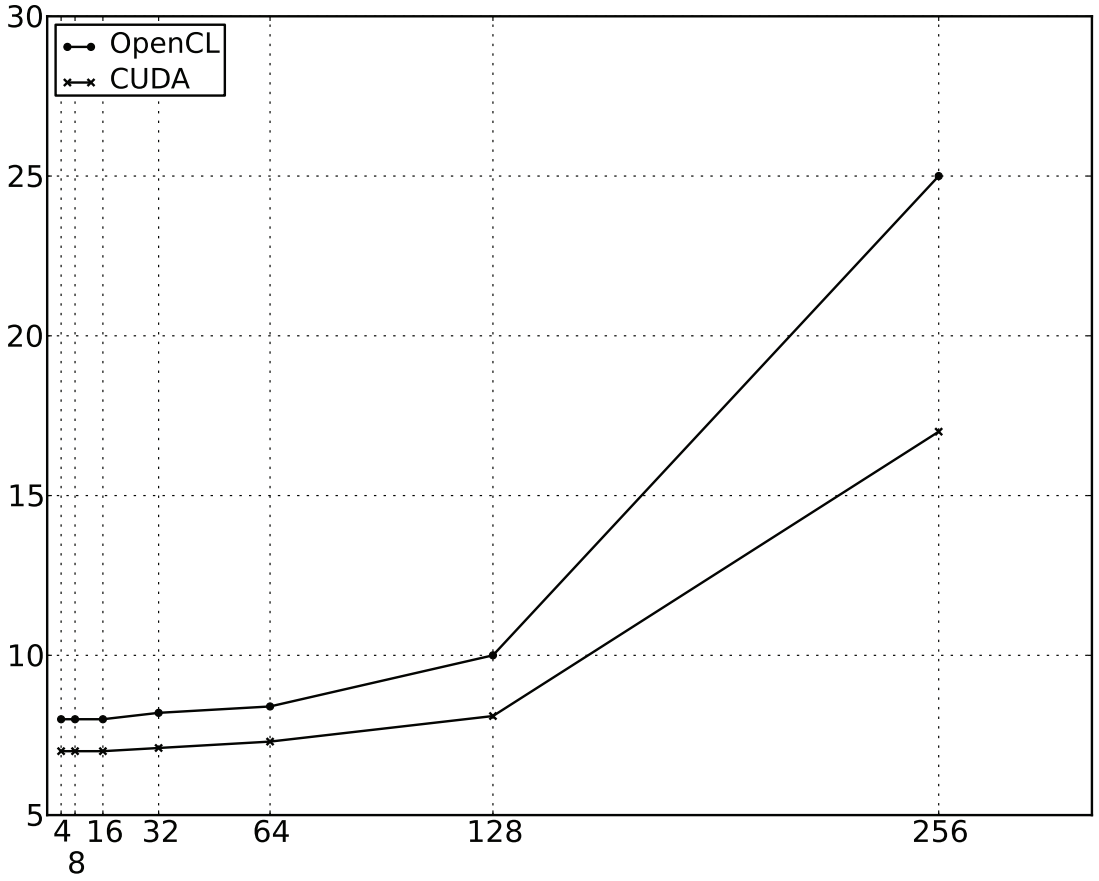

Рис. 5. Зависимость времени расчетов от размера ребра кубического «поля» при использовании CUDA и OpenCL