УДК 004.272.44

РЕАЛИЗАЦИЯ НЕЙРОСЕТЕВЫХ АЛГОРИТМОВ СРЕДСТВАМИ ВИДЕОКАРТЫ С ПОМОЩЬЮ ТЕХНОЛОГИИ NVIDIA CUDA

Шатохин Н.А.
Донецкий национальный технический университет

Введение

Многие задачи требуют вычислений с большим количеством операций, которые занимают значительные ресурсы даже современной технике, более того, можно с уверенностью считать, что каких бы скоростей ни достигла вычислительная техника всегда найдутся задачи, на решение которых потребовалось бы значительное время. Многие из таких сложных задач требуют, чтобы результат был получен за как можно меньшее время или даже строго ограниченное. К таким задачам, например, относится прогнозирование погоды, обработка изображений и распознание образов при управлении техникой. С другой стороны представляет большую техническую проблему уменьшение времени исполнения каждой операции в микропроцессоре.

Очевидным способом увеличить скорость вычислений было бы применение не одного вычислительного устройства, а нескольких, работающих совместно над решением одной задачи. Такой подход носит название параллельных вычислений. Несмотря на кажущуюся простоту решения, оно является подчас весьма нетривиальной задачей по проектированию вычислительной техники и разработки алгоритмов. Первая проблема кроется в том, что для того, чтобы задачу можно было решить с помощью параллельных вычислений алгоритм её решения должен допускать распараллеливание, мало того, далеко не каждая задача может быть

решена параллельным алгоритмом. Другой же, не менее важной проблемой является построение системы, на которой бы возможна была реализация параллельных вычислений.

Одним из способов параллельной обработки данных является использование искусственных нейронных сетей.

Нейронные сети и нейрокомпьютеры

Искусственные нейронные сети (ИНС) — математические модели, а также их программные или аппаратные реализации, построенные по принципу организации и функционирования биологических нейронных сетей — сетей нервных клеток живого организма. Это понятие возникло при изучении процессов, протекающих в мозге, и при попытке смоделировать эти процессы. Первой такой попыткой были нейронные сети Маккалока и Питтса [1]. Впоследствии, после разработки алгоритмов обучения, получаемые модели стали использовать в практических целях: в задачах прогнозирования, для распознавания образов, в задачах управления и др.

ИНС представляют собой систему соединённых и взаимодействующих между собой простых процессоров (искусственных нейронов). Такие процессоры обычно довольно просты, особенно в сравнении с процессорами, используемыми в персональных компьютерах. Каждый процессор подобной сети имеет дело только с сигналами, которые он периодически получает, и сигналами, которые он периодически посылает другим процессорам. И тем не менее, будучи соединёнными в достаточно большую сеть с управляемым взаимодействием, такие локально простые процессоры вместе способны выполнять довольно сложные задачи.

Нейрокомпьютер-этовычислительнаясистемасархитектурой аппаратного и программного обеспечения, адекватной выполнению алгоритмов, представленных в нейросетевом логическом базисе. Нейрокомпьютеры имеют MSIMD-архитектуру (Multiple SIMD),

т.е. это системы с параллельными потоками одинаковых команд и множественным потоком данных. Современные видеокарты имеют множество потоковых процессоров с SIMD-архитектурой, что позволяет им решать алгоритмы предназначенные для MSIMD. Для универсальных вычислений на графических чипах была разработана технология CUDA [2].

Технология NVIDIA CUDA

Технология CUDA — это программно-аппаратная вычислительная архитектура NVIDIA, основанная на расширении языка Си, которая даёт возможность организации доступа к набору инструкций графического ускорителя и управления его памятью при организации параллельных вычислений. CUDA помогает реализовывать алгоритмы, выполнимые на графических процессорах видеоускорителей GeForce восьмого поколения и старше (серии GeForce 8, 9 и GeForce 200), а также Quadro и Tesla.

Хотя трудоёмкость программирования GPU при помощи CUDA довольно велика, она ниже, чем с ранними GPGPU (General-purpose graphics processing units, GPU общего назначения) решениями. Такие программы требуют разбиения приложения между несколькими мультипроцессорами, но без разделения данных, которые хранятся в общей видеопамяти. И так как CUDA программирование для каждого мультипроцессора подобно OpenMP программированию (Open Multi-Processing, набор директив компилятора, библиотечных процедур и переменных окружения, которые предназначены для программирования многопоточных приложений на многопроцессорных системах с единой памятью на языках C, C++ и Fortran) [3], оно требует хорошего понимания организации памяти. Но, конечно же, сложность разработки и переноса на CUDA сильно зависит от приложения.

Целью работы является исследование возможностей технологии CUDA и определение ее пригодности для реализации нейросетевых алгоритмов.

Постановка задачи

Рассмотрим решение системы линейных алгебраических уравнений с помощью нейронных сетей.

В общем виде система линейных уравнений выглядит следующим образом:

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2 \\ \ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \end{cases}. \quad (1)$$

В матричной форме СЛАУ примет вид:

$$AX = B, \quad (2)$$

где A — матрица коэффициентов при X

$$A = \begin{Bmatrix} a_{11} & a_{12} & a_{13} & \ldots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \ldots & a_{2n} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{m1} & a_{m2} & a_{m3} & \ldots & a_{mn} \end{Bmatrix} \quad (3)$$

B — вектор свободных членов:

$$B = \begin{Bmatrix} b_1 \\ b_2 \\ \ldots \\ b_m \end{Bmatrix} \quad (4)$$

X — вектор неизвестных:

$$X = \begin{Bmatrix} x_1 \\ x_2 \\ \ldots \\ x_n \end{Bmatrix} \quad (5)$$

Необходимо построить нейросетевой алгоритм нахождения вектора X.

Нейроалгоритм

Входной сигнал нейросети — вектор X.

Выходной сигнал нейросети — некоторый вектор Y:

$$Y = \begin{Bmatrix} y_1 \\ y_2 \\ \cdots \\ y_m \end{Bmatrix}. \quad (6)$$

Каждая компонента y_i характеризует разницу между левой и правой частями соответствующих уравнений.

Желаемый вектор - $\overline{Y}$

$$\overline{Y} = \begin{Bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix}. \quad (7)$$

Функция активации:

$$f(x) = \begin{cases} \mathbf{0}, \ \mathbf{if} -\varepsilon < g_i < \varepsilon \\ g_i, \ \mathbf{else} \end{cases} \quad (8)$$

где ε — параметр, определяющий погрешность вычислений корней системы, задаваемый вручную.

Структура динамической системы решения СЛАУ приведена на рис. 1.

Структура нейросети приседена на рис. 2.

Результат

Было разработано три программы. В первой весь нейрон моделировался с помощью одного потока. Таким образом,

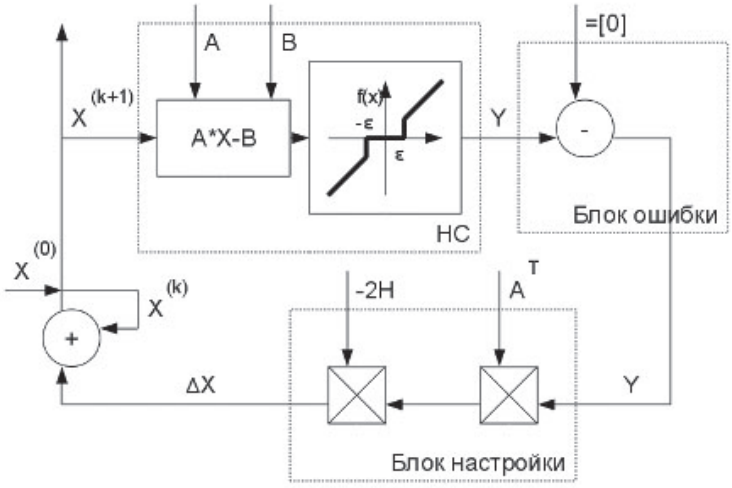

Рисунок 1 – Динамическая система для решения СЛАУ

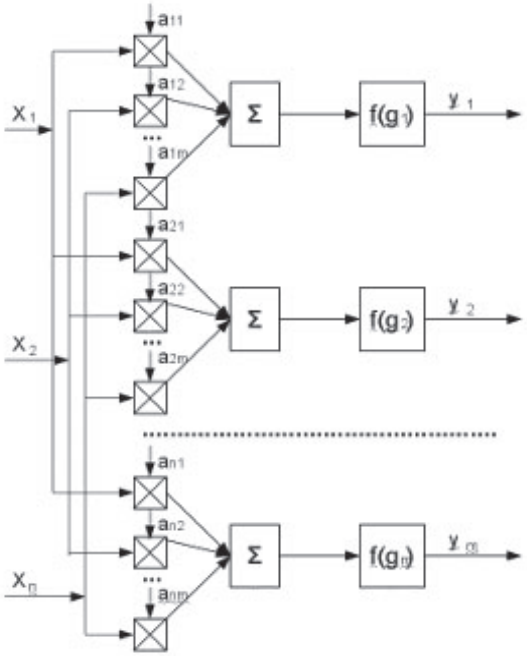

Рисунок 2 – Структура нейросети

сначала последовательно перемножались все элементы, затем они суммировались, и к результату применялась функция активации. Все нейроны работали параллельно (на графике GPU).

Во второй, был распараллелен также и процесс умножения. Нейрон здесь моделировался уже с помощью нескольких потоков (GPUop).

Третья программа была полностью последовательной и исполнялась на CPU (на графике CPU).

Были проведены эксперименты, в ходе которых было замерено время выполнения программы, в зависимости от размерности n системы. Результаты на рис. 3.

Как мы можем видеть, изначально наименьшее время показывает CPU, но его график растет быстрее остальных. Это связано с тем, что центральный процессор является более быстрым устройством, чем видеочип. Но благодаря способности выполнять множество потоков одновременно, графический процессор быстрее справляется с большимим объемами данных. Программа, которая была больше распараллелена (GPUpr) проигрывает в скорости более простой (GPU) из-за необходимости чаще обращаться к видеопамяти.

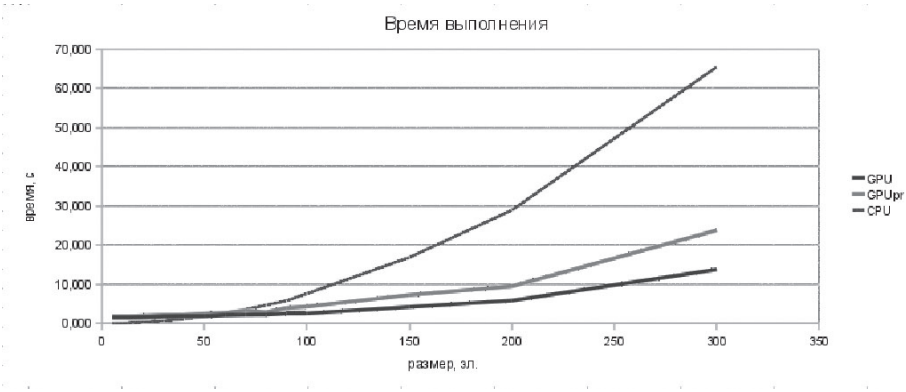

Рисунок 3 – Графики зависимости времени выполнения программы от размерности системы

Альтернативы

Альтернативой технологии CUDA является технология OpenCL. Она представляет собой средство разработки программного обеспечения, поддерживающее работу как с центральным процессором, так и видеокартами. Как следует из названия, OpenCL относится к семейству продуктов с открытым кодом, а потому не привязан к определенному производителю графических адаптеров. Работать с ним могут как устройства от ATI, так и NVIDIA и других компаний. Пока что модели на OpenCL находятся в демонстративной стадии, то есть сами производители видеокарт и процессоров показывают, на что способна данная среда разработки. Тем не менее будущее у продукта должно быть: на OpenCL сделала ставку компания Apple, планирующая использовать возможности видеокарт в программном обеспечении и новом поколении операционных систем. Скорее всего, этот открытый стандарт привлечет и немало других разработчиков.

Существует и ответ от ATI — среда программирования под названием ATI Stream, но по сравнению с CUDA она является намного более низкоуровневым языком программирования, что затрудняет написание приложений. Теоретическая вычислительная мощность видеокарт ATI, обычно измеряемая количеством операций с плавающей точкой в секунду — флопс, намного превосходит этот показатель у решений NVIDIA. Но из-за сложности создания программное обеспечение, использующее для вычислений видеокарты ATI, встречается намного реже.

Существуют и специализированные устройства для параллельных вычислений. Так, компания ClearSpeed серийно выпускает низкопотребляющие (около 10 Вт) PCI-X-платы вычислительных акселераторов с производительностью порядка 50 Гигафлопс, при этом способные работать с 64-битовыми числами с плавающей точкой (возможности CUDA сегодня ограничены 32-битовыми вычислениями, правда, конструкторы NVIDIA обещают в обозримом будущем 64-битовую аппаратную платформу).

Заключение

Технологии, позволяющие вычислять математические данные на графических процессорах, появились относительно недавно, но уже успели продемонстрировать свою высокую эффективность. Технология CUDA используется во многих областях науки, где есть необходимость в обработке огромных массивов данных, становясь дешевой альтернативой дорогим нейрокомпьютерам. С ее помощью можно относительно легко реализовать любой нейроалгоритм.

Литература

[1] Мак-Каллок У. С., Питтс В.,Логическое исчисление идей, относящихся к нервной активности // В сб.: «Автоматы» под ред. К. Э. Шеннона и Дж. Маккарти. — М.: Изд-во иностр. лит., 1956. — с.363-384. (Перевод английской статьи 1943 г.)

[2] Александр Перекалин. CUDA быстрее. // Журнал «CHIP», Июль 2009. — М.: ИД «Бурда», 2009. — с. 70-73

[3] Nawal Copty. Introducing OpenMP: A Portable, Parallel Programming API for Shared Memory Multiprocessors [Электронный ресурс], — Sun Microsystems, 2007. —.— Режим доступа: http://developers.sun.com/solaris/articles/omp-intro.html