УДК 004.032.26

ПАРАЛЛЕЛЬНАЯ ОБРАБОТКА ПОТОКА ДАННЫХ ИСКУССТВЕННЫМИ НЕЙРОННЫМИ СЕТЯМИ НА ПЛАТФОРМЕ CUDA

В.А. Колбасин
Кандидат технических наук
Кафедра системного анализа и управления
Национальный технический университет «Харьковский политехнический институт»
ул. Фрунзе, 21, г. Харьков, Украина, 61002
Контактный тел.: 050-860-76-21
E-mail: Vjacheslav_K@mail.ru

Запропоновано схему програмної реалізації штучної нейронної мережі (ШНМ) прямого розповсюдження на платформі масових паралельних обчислень CUDA

Ключові слова: штучні нейронні мережі, CUDA

Предложена схема программной реализации искусственной нейронной сети (ИНС) прямого распространения на платформе массовых параллельных вычислений CUDA

Ключевые слова: искусственные нейронные сети, CUDA

The software implementation of a backpropagation artificial neural network (ANN) on massive parallel computing platform CUDA is proposed

Key words: artificial neural network, CUDA

1. Введение

Искусственные нейронные сети (ИНС) широко применяются при решении практических задач, связанных с обнаружением и классификацией сигналов различной природы [1, 2]. В частности, ИНС успешно применяются для распознавания речи, текста, диагностики состояния технических средств. Общей проблемой при практическом использовании ИНС является их высокая требовательность к вычислительным ресурсам, что при традиционной программной реализации ИНС в ряде случаев делает невозможной обработку данных в режиме реального времени.

Для ИНС характерно, что каждый нейрон является, по сути своей, независимым вычислителем и все нейроны одного слоя могут работать параллельно. Поэтому для увеличения производительности систем, использующих ИНС, используются технологии параллельных вычислений.

Параллельная реализация ИНС может быть осуществлена как на типовых, так и на специализированных вычислительных устройствах. В качестве специализированных вычислительных устройств чаще всего применяются нейропроцессоры и микросхемы программируемых логических интегральных схем (ПЛИС). Для реализации ИНС на типовых вычислительных устройствах применяют технологии параллельной обработки данных в рамках одного процессора, сюда относятся в основном команды потоковых SIMD-расширений процессора (SSE), и технологии многопроцессорных и распределенных вычислений OpenMP и MPI. Существенным следствием использования данных технологий является значительное увеличение сложности и стоимости системы.

Появившаяся сравнительно недавно технология массовых параллельных вычислений CUDA (Computer Unified Device Architecture) [3, 4] позволяет использовать процессоры видеокарт для выполнения произвольных расчетов, обеспечивая при этом высокую производительность при достаточно низкой стоимости решения. Так, при равной производительности стоимость системы на базе платформы CUDA в среднем на порядок ниже стоимости вычислительного кластера. Но при этом высокая производительность на платформе CUDA достигается только для задач, представимых в модели вычислений SIMD (одна инструкция выполняется для многих данных) и удовлетворяющих ряду ограничений по использованию памяти.

В работах [5-6] было показано, что перенос реализации ИНС на платформу CUDA приводит к определенному увеличению производительности при обработке ИНС одного набора входных данных, и предложены схемы параллельной реализации ряда ИНС. Вместе с тем, в данных работах не рассматривались особенности обработки потока данных при помощи ИНС, хотя именно эта задача чаще всего возникает на практике. Поэтому целью данной работы является исследование возможности ускорения обработки потока данных посредством ИНС прямого распространения за счет распараллеливания обработки для платформы CUDA.

2. Обработка данных с использованием ИНС

В данной работе будем рассматривать подход к обработке, при котором входной поток данных разбивается на окна анализа фиксированной длины, а затем данные каждого окна передаются на вход ИНС. Если в задаче используется перекрытие окон анализа, то будем предполагать, что окна обрабатываются по отдельности и оптимизация обработки зоны перекрытия окон не производится.

Также ограничимся рассмотрением многослойной ИНС прямого распространения с сигмоидной активационной функцией и количеством нейронов выходного слоя, равным количеству нейронов входного слоя, поскольку такая ситуация является наиболее сложной с точки зрения производительности.

Значение на выходе каждого нейрона вычисляется по формуле [1]:

$$y_i = f\left(\sum_{j=0}^{N-1} x_j \cdot w_{i,j}\right), \quad (1)$$

где x_j - значение на j-м входе i-го нейрона;
$w_{i,j}$ - весовой коэффициент;
y_i - значение на выходе i-го нейрона;
$f(z) = 1/\left(1+e^{-x}\right)$ - функция активации.

Слои нейронов обрабатываются последовательно: сначала вычисляются значения на выходах всех нейронов первого слоя, затем, используя полученные значения как входные значения следующего слоя, вычисляются значения на выходах второго слоя и т.д. Значения на выходах нейронов последнего слоя рассматриваются в качестве результата обработки данных ИНС.

3. Оптимизация вычисления ИНС для CUDA

Платформа CUDA [3, 4] представляет собой многопроцессорный вычислитель, каждый процессор которого (потоковый мультипроцессор, SMP) является SIMD устройством, т.е. выполняет одну инструкцию над набором из нескольких операндов. Программно обработка каждого из операндов представляется отдельным потоком выполнения (Thread), но при этом все потоки одного SMP выполняют одну и ту же инструкцию. Инструкции могут не выполняться (не осуществлять изменений) для некоторых потоков, но не могут отличаться. Параллельное выполнение разных инструкций на одном SMP платформа не поддерживает, но для разных SMP такое допускается. Исполняемые на одном SMP потоки CUDA объединяются в блок, а все потоки, относящиеся к одной задаче – в сетку потоков (grid).

Платформа CUDA имеет несколько областей памяти: это глобальная память компьютера, память платформы (видеокарты), кэш доступа к памяти платформы и так называемая разделяемая память (shared memory). Виды памяти перечислены в порядке уменьшения времени доступа к ней. Кэширование осуществляется при доступе к областям памяти, выделенным для хранения констант и текстур. Объем кэша зависит от платформы, но не превышает 8Кб на один SMP. Разделяемая память размещается на SMP и обеспечивает минимальное время доступа, но имеет объем в 16 Кб и разделена на банки. Наибольшая производительность при работе с разделяемой памятью достигается, когда каждый поток обращается к своему банку памяти.

Так как нейрон представляет собой независимое вычислительное устройство, то наиболее простой представляется параллельная реализация ИНС, в которой каждый поток выполнения будет вычислять значение на выходе отдельного нейрона. В этом случае обеспечивается минимальное время обработки одного окна данных, но существенно возрастают накладные расходы, связанные с обменом данных с аппаратурой CUDA и запуском процесса обработки окна данных. При использовании этого подхода для трехслойной нейронной сети при изменении длин окон анализа в диапазоне 50 – 400 отсчетов не удалось достичь скорости обработки данных больше $2 \cdot 10^6$ отсчетов в секунду. Что сравнимо со скоростью работы программной реализации ИНС на обычном процессоре.

Учитывая, что разделяемая память является локальной для каждого SMP, и в рамках разных SMP возможно параллельно выполнять различные инструкции предлагается выполнять обработку ИНС одного окна исходных данных на одном SMP, т.е. в терминологии CUDA – внутри одного блока потоков выполнения. Иными словами – предлагается перейти от модели параллельной обработки одного окна анализа одной ИНС к модели параллельной обработки нескольких окон анализа несколькими одинаковыми ИНС. При этом появляется возможность программно запланировать выполнение большего числа блоков, чем количество фактически установленных на аппаратуре SMP – блоки будет выполняться последовательно, но уменьшатся затраты на их запуск.

С учетом этого, предлагается выполнять обработку потока данных по такой схеме:

1. Накапливается набор из $N \cdot G$ отсчетов входных данных, где N – длина окна анализа, а G – количество обрабатываемых за один раз окон анализа.

2. Накопленные данные копируются в память устройства CUDA и запускается их обработка с помощью ИНС.

3. После завершения обработки, результаты обработки копируются в память компьютера.

4. Действия 1-3 повторяются.

При вычислении значения на выходе нейрона по формуле (1) весовые коэффициенты входов нейронов будут участвовать в обработке одного окна данных единожды. Значения, подаваемые на вход слоя ИНС, напротив, будут использоваться много раз всеми нейронами этого слоя. Кроме того, для хранения значений входов слоя ИНС требуется существенно меньше памяти, чем для хранения весов входов нейронов. Поэтому значения входов и выходов слоя ИНС имеет смысл разместить в разделяемой памяти. А для достижения максимально возможного ускорения доступа к весам входов нейронов – их чтение имеет смысл осуществлять с использованием кэша текстур.

Фрагмент соответствующего кода, считывающего исходные данные окна анализа в разделяемую память и вычисляющего значения на выходе первого слоя ИНС, приведен в листинге 1.

```
// Буфер для хранения значений входов и выходов слоев ИНС
__shared__ float buf1[512], buf2[512];
// Выровненное на границу 64 байт смещение строк матрицы весов
// inCnt содержит размер окна анализа/количество нейронов в слое
int ofs = (inCnt + 15) & 0xFFF0;
// Адреса, по которым расположены входные и выходные значения блока
in += blockIdx.x*ofs;      out += blockIdx.x*ofs;

// Параллельная загрузка входных данных в разделяемую память
int i=0;
while(i < inCnt){
        buf[i + threadIdx.x] = in[i + threadIdx.x];
        i += BLOCK_SIZE;
};
__syncthreads ();

// Нахождение значений выходов нейронов данного слоя
for (int output=0; output<inCnt; output += BLOCK_SIZE){
        int p2 = output + threadIdx.x;
        if (p2 < inCnt){
                float res = 0;
                for (int input=0; input<inCnt; input++)
                        res += buf[input]* matr[p2 + input*ofs];;
                buf[p2] = 1/(1+__expf(-res));
        };
};
__syncthreads ();
```

Листинг 1. Фрагмент кода загрузки данных и обработки слоя ИНС.

4. Результаты

Производительность предложенной схемы обработки данных была измерена для различных длин окон анализа при использовании разных режимов параллельной реализации ИНС. Для тестирования производительности использовался компьютер следующей конфигурации: центральный процессор Intel Core 2 Duo, работающий на частоте 2,1 ГГц, видеокарта GeForce 8800GT, поддерживающая технологию CUDA, на которой установлено 14 потоковых мультипроцессоров.

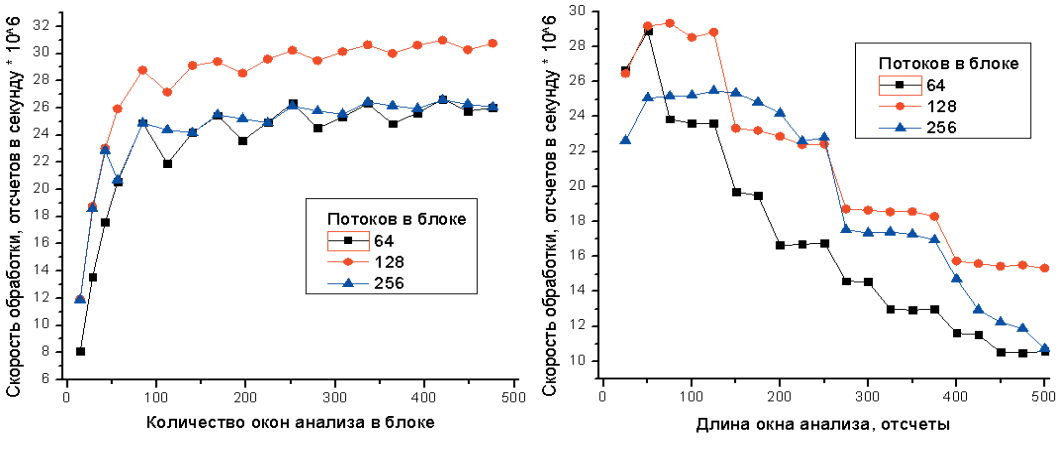
Зависимости скорости обработки потока данных с использованием предложенной реализации на платформе CUDA трехслойной ИНС с одинаковым числом нейронов в каждом слое от длины окна анализа и количества потоков в блоке представлены на рис. 1.

Как видно из приведенных графиков, скорость обработки вначале быстро увеличивается с ростом количества окон анализа, обрабатываемых в блоке. Но затем, при достижении уровня более 8 блоков на SMP, рост скорости замедляется из-за уменьшения относительного вклада накладных расходов на копирование данных и запуск вычислений на платформе CUDA в общее время вычислений ИНС.

Зависимость скорости обработки данных от длины окна анализа имеет ступенчатый характер и изменяется кратно количеству потоков выполнения в блоке. Если длина окна анализа больше количества одновременно выполняющихся в блоке потоков, то каждый поток выполняет расчет значения на выходе не одного, а нескольких нейронов, что соответственно увеличивает время обработки окна данных. При этом изменение длины окна анализа, не приводящее к увеличению числа итераций, слабо влияет на скорость обработки данных.

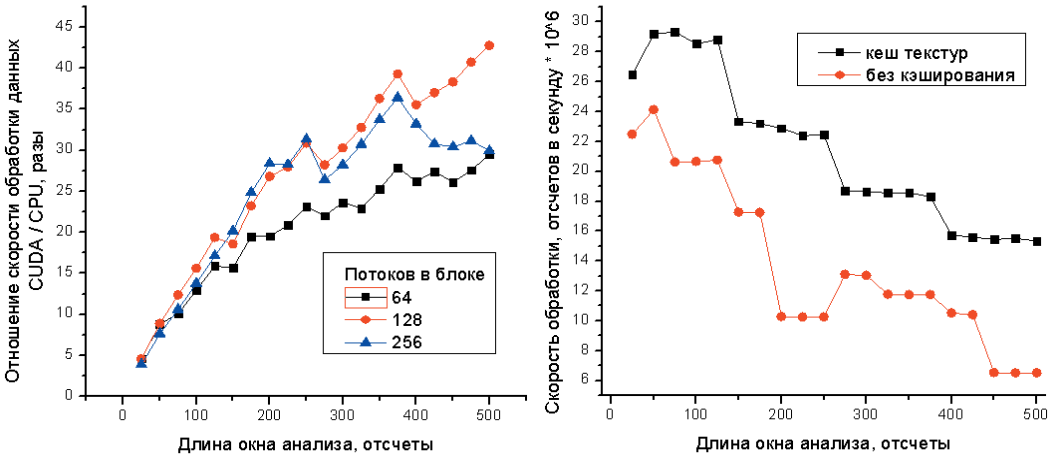
Для использованной видеокарты наибольшую производительность обеспечивает выбор 128 потоков выполнения в блоке.

Зависимости прироста производительности при использовании реализации на базе CUDA по сравнению с реализацией на базе традиционной многопоточности центрального процессора (CPU) и влияние использования текстурного кеша на скорость обработки данных приведены на рис. 2.



а) б)

Рис. 1. Зависимость скорости обработки данных трехслойной ИНС от количества окон в блоке (а) и от длины окна анализа при 196 окнах анализа в блоке (б)



а) б)

Рис. 2. Увеличение скорости обработки данных при использовании платформы CUDA по сравнению с использованием CPU (а) и влияние текстурного кеша на скорость обработки данных (б)

Как видно из приведенных графиков использование кеша текстур в ряде случаев позволяет ускорить обработку данных практически в два раза за счет уменьшения задержек доступа к памяти для чтения входных весов нейронов.

В целом при использовании реализации ИНС на базе платформы CUDA скорость обработки данных для окон анализа длиной более 200 отсчетов увеличивается в 30 раз по сравнению скоростью обработки на двухядерном процессоре. Для малых длин окон анализа прирост производительности оказывается меньшим, но здесь возможна дальнейшая оптимизация за счет хранения весов входов нейронов в разделяемой памяти.

Выводы

Предложенная схема параллельной реализации на платформе CUDA ИНС прямого распространения обеспечивает в 30 раз большую скорость обработки потока данных по сравнению с реализацией на центральном процессоре при длине окна анализа более 200 отсчетов. При выборе окна анализа меньшей длины увеличение скорости обработки оказывается меньше, но в этом случае возможна дальнейшая оптимизация за счет размещения части весовых коэффициентов нейронов в разделяемой памяти.

Данная схема может быть использована в задачах обработки данных в режиме реального времени, использующих аппарат искусственных нейронных сетей прямого распространения.

Литература

1. Бодянский, Е.В. Искусственные нейронные сети [Текст] / О. Г. Руденко, Е. В. Бодянский. - Х.: Компания СМИТ, 2005. - 408 с.
2. Осовский, С. Нейронные сети для обработки информации [Текст] / С. Осовский. – М. Финансы и статистика, 2004. – 344 с.
3. NVidia CUDA Programming Guide [Электронный ресурс] / NVidia Corp, 2008. – Режим доступа: http://developer.download.nvidia.com/compute/cuda/3_2/toolkit/docs/CUDA_C_Progra-ming_Guide.pdf. - 10.05.2011 г. - Загл. с экрана.
4. Боресков, А.В. Основы работы с технологией CUDA [Текст] / А.В. Боресков, А.А. Харламов. – М.:ДМК Пресс, 2010. – 232 с.
5. Jang, H. H. Neural Network Implementation Using CUDA and OpenMP [Текст] / H.H. Jang, A.J. Park, K.C. Jung // Proceeding of Computing: Techniques and Applications, 2008. - p. 155–161.
6. Uetz, R. Large-scale Object Recognition with CUDA-accelerated Hierarchical Neural Networks Intelligent Computing and Intelligent Systems / R. Uetz, S. Behnke // Proceeding of Intelligent computing and Intelligent Systems, 2009. – p. 536 – 541.