Нижегородский государственный университет им. Н.И. Лобачевского

Факультет Вычислительной математики и кибернетики

Реализация и оптимизация матричного произведения

Бастраков С.И.

ВМК ННГУ

bastrakov@vmk.unn.ru

Молодежная научная школа «Суперкомпьютерные технологии и высокопроизводительные вычисления в образовании, науке и промышленности» 5 – 8 ноября 2014 г., г. Нижний Новгород

Задание 1

- Реализовать на CUDA функцию mmult_gpu, перемножающую две прямоугольные матрицы.
- Размер матриц всегда будет кратен BLOCK_SIZE (=16).
- Указание: пусть каждый поток считает один элемент результирующей матрицы. Используйте двумерные блоки размера BLOCK_SIZE x BLOCK_SIZE и двумерную сетку блоков.
- Убедиться, что полученный результат совпадает с результатом CPU-версии.
- Обратите внимание: “Test passed.” должно появиться в 5й сверху строке, а не последней – последняя строка вывода относится к еще не реализованной оптимизированной версии.

Анализ

- Пусть выполняемые блоки потоков имеют размер BLOCK_SIZE на BLOCK_SIZE, размер матрицы A — m на n, а матрицы B — n на k.
- Проанализируем доступы к глобальной памяти для одного блока потоков.
- Каждый поток считывает одну строку матрицы A и один столбец матрицы B, т.е. $2n$ чисел. Итого блок считывает $2n$ * BLOCK_SIZE^2 чисел.
- Но ведь на самом деле для подсчёта подматрицы C BLOCK_SIZE на BLOCK_SIZE требуется BLOCK_SIZE строк из A, и BLOCK_SIZE столбцов из B, то есть только $2n$ * BLOCK_SIZE чисел?

Задание 2

- Реализуйте блочное произведение матриц
- Блок потоков с номером (I, J) вычисляет подматрицу C_{IJ}:

$$\begin{pmatrix} A_{I1} & A_{I2} & \cdots & A_{IN} \end{pmatrix} \times \begin{pmatrix} B_{1J} \\ B_{2J} \\ \vdots \\ B_{NJ} \end{pmatrix} = \begin{pmatrix} C_{IJ} \end{pmatrix}$$

$$C_{IJ} = \sum_{p=0}^{N} A_{Ip} \times B_{pJ}$$

Задание 2

- Выделите в разделяемой памяти два массива A’ и B’ размера BLOCK_SIZE на BLOCK_SIZE.
- Пусть поток, соответствующий элементу (i; j) в матрице $C_{IJ,}$ действует по следующему алгоритму:
  - для каждого p:
    - скопировать элемент (i; j) из A_{Ip} в A’
    - скопировать элемент (i; j) из B_{pJ} в B’
    - __syncthreads
    - посчитать произведение i-й строки A’ и j-й строки B’
    - __syncthreads
  - записать сумму произведений в элемент (i; j) C_{IJ}
- Реализовать функцию mmult_gpu_opt, использующую предложенную блочную схему, и убедиться, что результат верный (”Test passed.” в последней строке вывода).

Дополнительное задание 1

- Модифицировать блочную схему так, чтобы каждый поток вычислял не один, а несколько элементов
  - Таким образом, размер блока матрицы будет кратен размеру блока потоков (а не равен ему, как в оригинальной блочной схеме)
- Подберите оптимальную конфигурацию размера блока потоков и блока матрицы для матрицы из теста

Дополнительное задание 2

- Реализуйте алгоритм Штрассена (описание можно найти, например, в Википедии)
- Для простоты можно считать, что размеры матриц являются степенями 2. В таком случае необходимо исправить их в начале функции main
- Используйте разработанное ранее ядро для перемножения подматриц
- Эмпирически подберите соотношение на размеры для матриц, при котором имеет смысл умножать по определению
- Сравните производительность блочного алгоритма и алгоритма Штрассена на матрицах различных размеров