Курс «Параллельное программирование»

Лабораторная работа №3. Параллельное вычисление произведения матриц с использованием OpenMP

Юлдашев Артур Владимирович
art@ugatu.su
Спеле Владимир Владимирович
spele.vv@ugatu.su

Кафедра высокопроизводительных вычислительных технологий и систем (ВВТиС)

Цель работы

Приобрести навыки распараллеливания вложенных циклов с использованием директив OpenMP. Исследовать ускорение, эффективность и производительность многопоточных реализаций алгоритмов решения задачи матричного умножения.

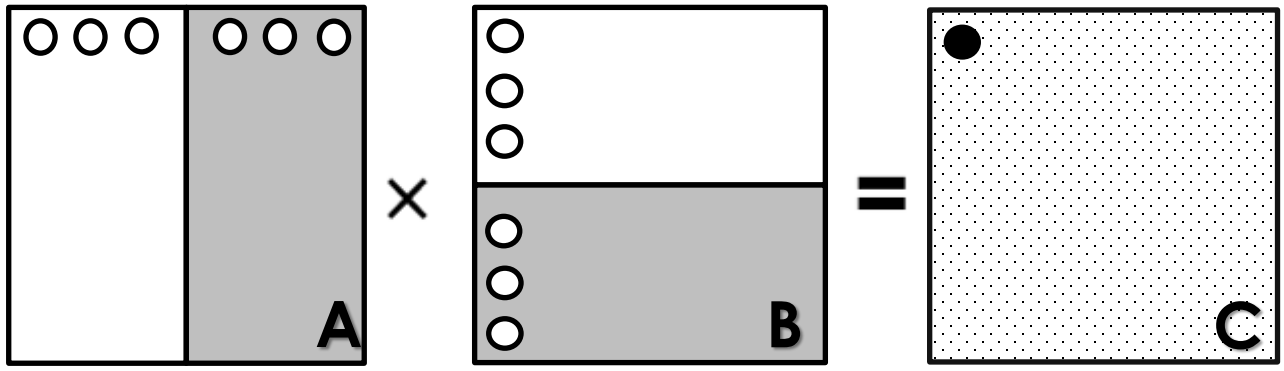
Вычисление произведения двух матриц

Требуется перемножить квадратные матрицы и вычислить квадрат евклидовой нормы результирующей матрицы:

$$C = AB, \qquad \|C\|^2 = \sum_{i=1}^{N} \sum_{j}^{L} c_{ij}^2$$

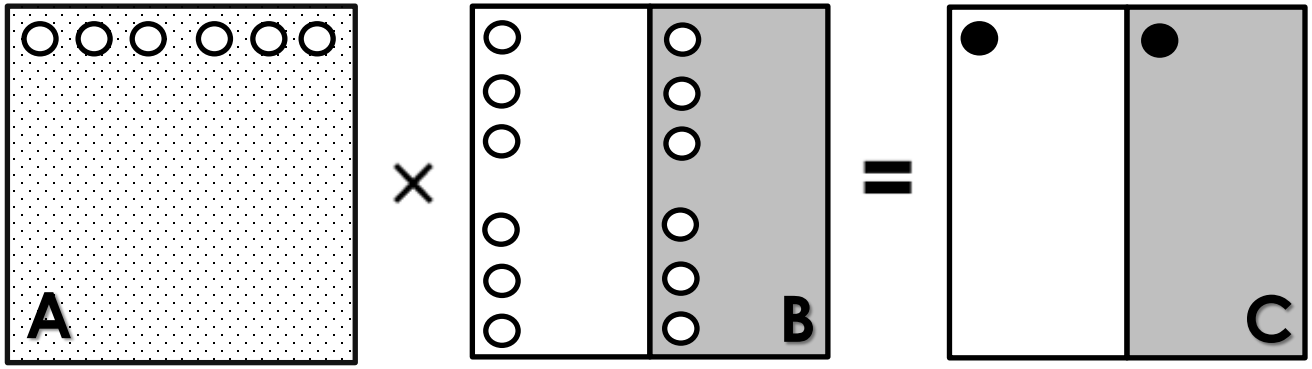
Матричное умножение выражается формулой

$$c_{ij} = \sum_{k=1}^{N} a_{ik} b_{kj}\,, \qquad i, j = \overline{1 \ldots N},$$

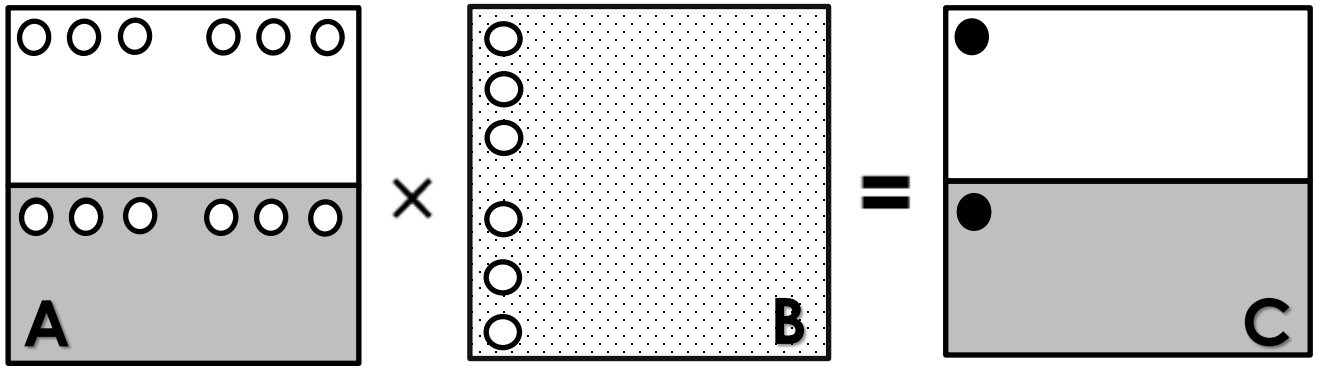


Параллельное умножение матриц (алгоритм №1)

Декомпозиция по столбцам или по строкам



Параллельное умножение матриц (алгоритм №2)



Параллельное умножение матриц (алгоритм №3)

Анализ полученных результатов

Определение. Отношение времени выполнения параллельной программы на одном процессоре (ядре) T_1^* ко времени выполнения параллельной программы на p процессорах T_p называется **ускорением** при использовании p процессоров:

$$S_p^* = \frac{T_1^*}{T_p}$$

Определение. Отношение ускорения S_p^* к количеству процессоров p называется **эффективностью** при использовании p процессоров:

$$E_p^* = \frac{S_P^*}{p}$$

Анализ полученных результатов

Определение. Пиковой (теоретической) производительностью называется максимальное количество команд или операций, которое может теоретически выполнять вычислительная система в единицу времени при условии постоянной и полной загрузки всех ее исполнительных устройств.

Пиковая производительность чаще всего измеряется в количестве выполняемых в секунду операций с плавающей точкой – Flops (Floating point operation per second). Формула для расчета пиковой производительности R_{peak} имеет вид:

$$R_{peak} = pnv,$$

где

p – количество процессоров или ядер вычислительной системы;

n – теоретическое количество операций с плавающей точкой, которое может выполнять процессор или ядро за 1 такт;

v – тактовая частота, на которой работает процессор (ядро).

Анализ полученных результатов

Определение. Реальной производительностью называется количество команд или операций, которое выполняет вычислительная система в единицу времени для конкретного алгоритма(программы).

Чтобы вычислить один элемент результирующей матрицы C, необходимо выполнить скалярное умножения строки матрицы A на столбец матрицы B – требуется совершить N операций умножения и столько же операций сложения. Тогда для вычисления полной матрицы C требуется $2N^3$ вещественных операций.

Тогда если за время T осуществляется q матричных умножений, получим, что реальную производительность можно выразить формулами:

$$R^1_{real} = \frac{2qN^3}{T_1} \qquad R^p_{real} = \frac{2qN^3}{T_p}$$

Чтобы оценить эффективность использования вычислительной системы при выполнении последовательного и параллельного алгоритмов, введем коэффициенты:

$$U_1 = \frac{R^1_{real}}{R_{peak}} \qquad U_p = \frac{R^1_{real}}{R_{peak}}$$

Задание

1. Выполнить программную реализацию на языке C/C++ последовательного алгоритма умножения двух квадратных матриц размера $N \times N$. Предусмотреть:

   1) ввод количества повторов умножения q пользователем с клавиатуры;
   2) статическое выделение памяти для хранения матриц;
   3) заполнение матриц случайными вещественными числами в диапазоне от -0.5 до 0.5;
   4) повторение процедуры умножения q раз;
   5) вывод на экран квадрата евклидовой нормы результирующей матрицы и времени работы программы.

   Время замерять при помощи функции *omp_get_wtime*.
2. Выполнить распараллеливание матричного умножения путем добавления директив OpenMP в текст последовательной программы. Реализовать три алгоритма параллельного умножения, предложенные ранее.
3. Отладить написанные программы при небольших размерностях матриц N на многоядерной вычислительной системе. Убедиться, что результат работы параллельных программ совпадает с полученным в последовательной версии и является стабильным при многократных запусках.

Задание

4. Запустить все три параллельные версии при размерности матриц $N = 100$ и количестве повторов q таком, что время работы каждой из программ составляло бы порядка 10 секунд. Выбрать наиболее производительную версию для дальнейшей работы.
5. Запустить последовательную программу при различных размерностях матриц $N = 5, 10, 50, 100, 200, 500$, выбирая q так, чтобы время работы программы составляло не менее 10 секунд. Запустить параллельную программу при аналогичных значениях N и q. Время работы каждой программы занести в таблицу:

N	q	T_1	T_p
5			
10			
50			
100			
200			
500			

Задание

6. Вычислить ускорение и эффективность параллельной программы для каждого N, полученные значения занести в таблицу.

N	S	E
5		
10		
50		
100		
200		
500		

7. Построить графики зависимости ускорения и эффективности параллельной программы от размерности умножаемых матриц. Объяснить полученные результаты.

Задание

8. Определить пиковую производительность одного ядра R_1 и всей многоядерной системы R_p, на которой производятся вычисления.
9. Вычислить количество вещественных операций, выполняемых при матричном умножении для каждого N, реальную производительность, достигнутую в последовательной (R^1_{real}) и параллельной (R^p_{real}) версиях для каждой размерности и отношения реальной производительности к пиковой U_1 и U_p .
   Полученные значения занести в таблицу:

N	R^1_{real}	R^p_{real}	U_1	U_p
5				
10				
50				
100				
200				
500				

10. Построить график зависимости U_1 и U_p от размерности умножаемых матриц. Объяснить полученные результаты.

Задание

11. Определить размерность матриц N_{min} такую, что при любом $N < N_{min}$ параллельная версия программы работает медленнее, чем последовательная, а при $N > N_{min}$ имеется ускорение при использовании параллельной программы. Для определения N_{min} провести дополнительные запуски обеих версий программ.

12. Модифицировать параллельную программу путем добавления спецификатора if директивы parallel for так, чтобы порождение параллельной области происходило только при размерности матриц $N > N_{min}$. Убедится в корректности работы полученной программы.

Требования к оформлению отчета

- В отчет по проделанной работе включить:
  1) характеристики центрального процессора;
  2) цель работы с краткой формулировкой ожидаемого результата;
  3) постановку задачи;
  4) описание параллельного алгоритма и особенности его программной реализации;
  5) таблицы и графики с результатами вычислений и измерений;
  6) анализ полученных в ходе работы результатов, который предполагает их объяснение с помощью теоретических выкладок;
  7) вывод;
  8) приложение с листингом программ.