МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«СЕВАСТОПОЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт информационных технологий

кафедра «Информационные системы»

Пояснительная записка

к расчетно-графической работе

по дисциплине «Технологии распределенных систем и параллельных вычислений»

на тему: Параллельные методы матричного умножения.

Выполнили: студенты 4 курса, группы: ИС/б-20-1-о

Направления подготовки (специальности) 09.03.02 Информационные системы и технологии

профиль (специализация) Информационные системы и технологии

Лысенко И.Д.

Защита « » \_\_\_\_\_\_\_\_\_ 20 \_\_\_ г. Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (инициалы, фамилия)

Севастополь 2023 г.

**Постановка задачи**

В соответствии с вторым типом алгоритма ленточной схемы перемножения матриц выполнить перемножение матриц размером 5х5. При этом решение должно содержать детальное изложение алгоритма с указанием индексов перемножаемых каждым процессом элементов и с указанием видов данных (значений элементов матриц), обмен которыми реализуется в процессе выполнения алгоритма.

**Ход работы**

Алгоритм перемножения матриц размером 5x5 с использованием ленточной схемы в топологии "кольцо" и с обменом данными между 4 процессами может быть описан следующим образом:

* Инициализация матрицы C размером 5x5, которая будет содержать результат перемножения матриц A и B.
* Разделение процессов на 4 процесса, представляющих топологию "кольцо". Каждый процесс имеет свой уникальный идентификатор (номер процесса) от 1 до 4.
* Каждый процесс выполняет следующие действия:

1. Получает одну строку матрицы A и сохраняет ее в локальной памяти.
2. Получает одну строку матрицы B и сохраняет ее в локальной памяти.
3. Вычисляет промежуточные результаты перемножения элементов полученных строк матриц A и B, сохраняя промежуточные значения.
4. Обменивается промежуточными результатами с соседними процессами. Каждый процесс передает свои результаты следующему процессу в кольцевой топологии, а затем принимает результаты от предыдущего процесса. Обмен данными осуществляется путем передачи числовых значений (элементов матриц) между процессами.
5. Вычисляет итоговые значения элементов матрицы C путем сложения промежуточных результатов.

* Повторяет шаги 3.3-3.5 для каждой строки матрицы A и B.
* После завершения перемножения всех строк матриц A и B каждый процесс будет содержать свою соответствующую строку матрицы C, которая представляет собой результат перемножения матриц A и B.

Рассмотрим процедуру алгоритма детальнее.

Процесс 1:

* Получает строку A[1] матрицы A и сохраняет ее в локальной памяти.
* Получает строку B[1] матрицы B и сохраняет ее в локальной памяти.
* Вычисляет промежуточные результаты C[1][1], C[1][2], C[1][3], C[1][4], C[1][5] путем перемножения элементов A[1] и B[1].
* Обменивается промежуточными результатами C[1][1], C[1][2], C[1][3], C[1][4] с процессом 2.
* Принимает промежуточные результаты C[2][1], C[2][2], C[2][3], C[2][4] от процесса 2.
* Вычисляет итоговые значения C[1][1], C[1][2], C[1][3], C[1][4], C[1][5] путем сложения промежуточных результатов с полученными значениями от процесса 2.

Процесс 2:

* Получает строку A[2] матрицы A и сохраняет ее в локальной памяти.
* Получает строку B[2] матрицы B и сохраняет ее в локальной памяти.
* Вычисляет промежуточные результаты C[2][1], C[2][2], C[2][3], C[2][4], C[2][5] путем перемножения элементов A[2] и B[2].
* Продолжение детального описания алгоритма для каждого процесса:
* Процесс 2 (продолжение):
* Обменивается промежуточными результатами C[2][1], C[2][2], C[2][3], C[2][4] с процессом 1.
* Принимает промежуточные результаты C[3][1], C[3][2], C[3][3], C[3][4] от процесса 3.
* Вычисляет итоговые значения C[2][1], C[2][2], C[2][3], C[2][4], C[2][5] путем сложения промежуточных результатов с полученными значениями от процесса 1.

Процесс 3:

* Получает строку A[3] матрицы A и сохраняет ее в локальной памяти.
* Получает строку B[3] матрицы B и сохраняет ее в локальной памяти.
* Вычисляет промежуточные результаты C[3][1], C[3][2], C[3][3], C[3][4], C[3][5] путем перемножения элементов A[3] и B[3].
* Обменивается промежуточными результатами C[3][1], C[3][2], C[3][3], C[3][4] с процессом 2.
* Принимает промежуточные результаты C[4][1], C[4][2], C[4][3], C[4][4] от процесса 4.
* Вычисляет итоговые значения C[3][1], C[3][2], C[3][3], C[3][4], C[3][5] путем сложения промежуточных результатов с полученными значениями от процесса 2.

Процесс 4:

* Получает строку A[4] матрицы A и сохраняет ее в локальной памяти.
* Получает строку B[4] матрицы B и сохраняет ее в локальной памяти.
* Вычисляет промежуточные результаты C[4][1], C[4][2], C[4][3], C[4][4], C[4][5] путем перемножения элементов A[4] и B[4].
* Обменивается промежуточными результатами C[4][1], C[4][2], C[4][3], C[4][4] с процессом 3.
* Принимает промежуточные результаты C[1][1], C[1][2], C[1][3], C[1][4] от процесса 1.
* Вычисляет итоговые значения C[4][1], C[4][2], C[4][3], C[4][4], C[4][5] путем сложения промежуточных результатов с полученными значениями от процесса 3.

После выполнения алгоритма, каждый процесс будет содержать свою соответствующую строку матрицы C. Например, процесс 1 будет иметь строку C[1], процесс 2 - строку C[2], и так далее. Эти строки могут быть объединены в одну матрицу C для получения итогового результата перемножения матриц A и B. Обмен данными между процессами осуществляется при передаче промежуточных результатов и происходит в соответствии с топологией "кольцо", где каждый процесс передает результаты следующему процессу и принимает результаты от предыдущего процесса.

**Заключение**

Алгоритм перемножения матриц размером 5x5 с использованием ленточной схемы и топологии "кольцо" предоставляет эффективный способ распараллеливания вычислений между несколькими процессами. Каждый процесс выполняет перемножение строк матрицы A и строк матрицы B, обмениваясь промежуточными результатами с соседними процессами в кольцевой топологии.

Алгоритм обеспечивает минимальное количество обменов данными между процессами, так как каждый процесс передает и принимает только промежуточные результаты, относящиеся к его соседним процессам. Это позволяет снизить общую задержку и повысить эффективность вычислений.

Использование ленточной схемы позволяет уменьшить объем передаваемых данных между процессами, так как каждый процесс работает только с одной строкой матрицы A и B одновременно. Это уменьшает требования к памяти и общую сложность вычислений.

**СПИСОК ЛИТЕРАТУРЫ**

1. Богачев К.ЮОсновы параллельного программированияМ.: БИНОМ. Лаборатория знаний, 2003
2. Воеводин В.В., Воеводин Вл.ВПараллельные вычисленияСПб.: БХВ-Петербург, 20023.
3. Гергель В.П., Стронгин Р.ГОсновы параллельных вычислений для многопроцессорных вычислительных системН. Новгород: Изд-во ННГУ, 2001
4. Немнюгин С., Стесик ОПараллельное программирование для многопроцессорных вычислительных системСПб.: БХВ-Петербург, 2002
5. Andrews G.RFoundations of Multithreading, Parallel and Distributed ProgrammingAddison-Wesley, 2000