Отчёт по лабораторной работе №2

Ковшарев, Шишкин, Кошкин @ 20ПИ-2

1. Умножение матрицы на вектор с разбиением по строкам/столбцам

Краткое описание функций и алгоритмов:

initializeArrays: Инициализирует массивы для матрицы, вектора и результата. Эта функция подготавливает необходимые структуры данных для хранения матрицы, вектора, который будет умножаться на матрицу, и массива для хранения результата умножения.

matrixVectorMultiplication: Выполняет умножение матрицы на вектор. Эта функция принимает матрицу, её размеры (количество строк и столбцов), вектор и массив для сохранения результата. Реализует алгоритм умножения с разбиением по столбцам.

resetVector: Сбрасывает вектор в начальное состояние. Используется для очистки и подготовки векторов к повторному использованию в различных вычислениях.

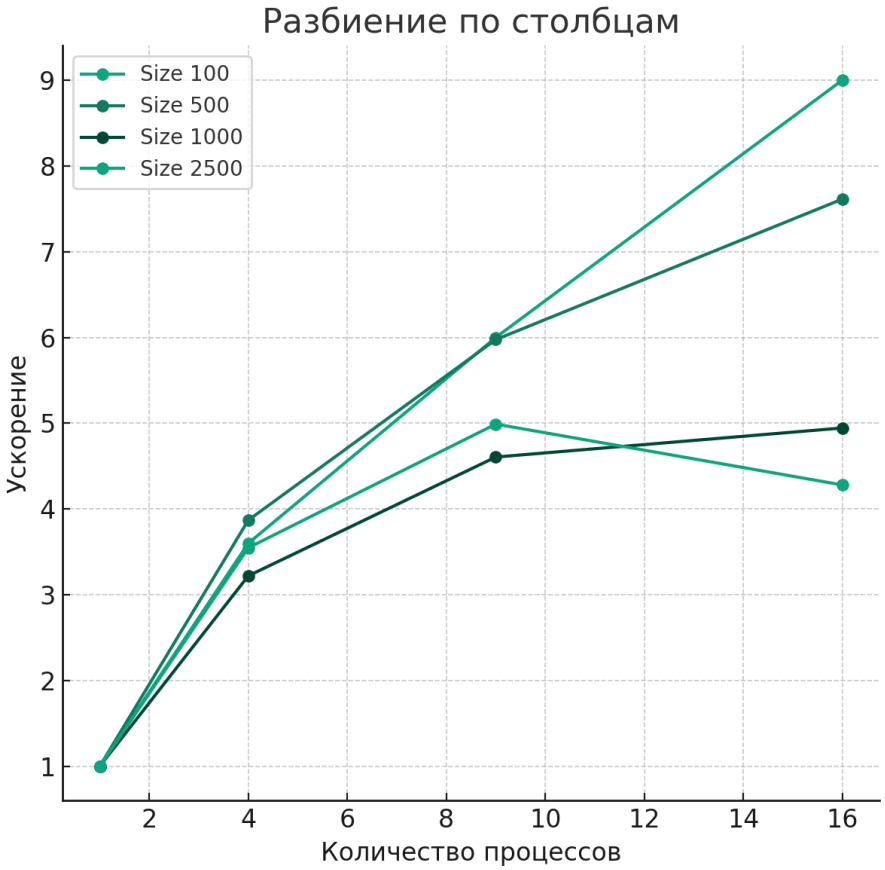
main: Главная функция программы, инициализирующая MPI (Message Passing Interface). Координирует выполнение программы, включая обработку аргументов командной строки, распределение задач между процессами, вызов других функций для выполнения основных операций, сбор и агрегацию результатов, а также замер времени выполнения операций. Завершает работу с MPI, обеспечивая корректное завершение параллельных процессов.

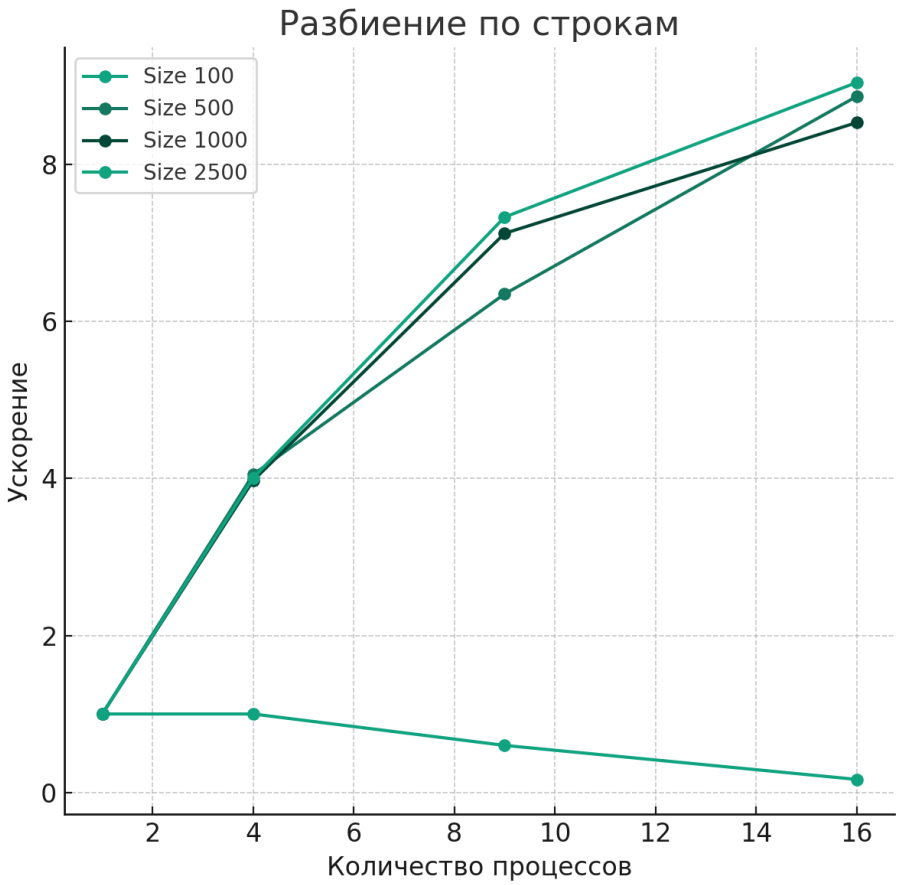
Формат ввода/вывода:

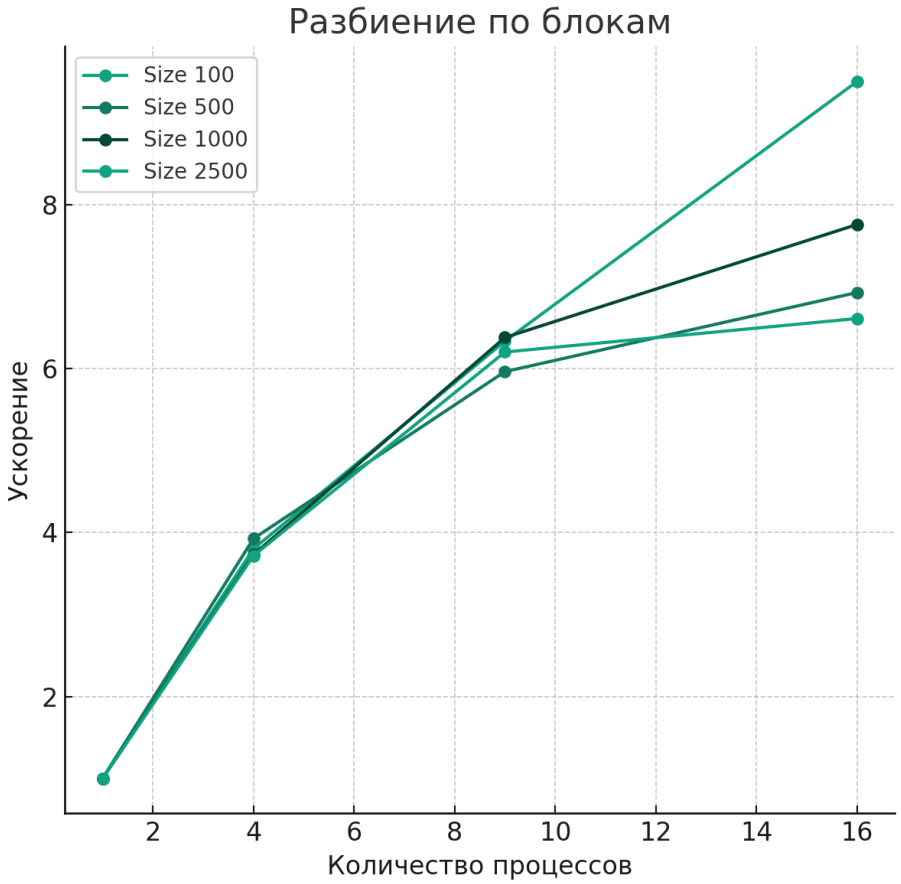
Ввод: Размер матрицы задается в коде (в данном случае, N = 2500), её содержимое зануляется, количество процессов определяется через MPI.

Вывод: Время выполнения операции умножения матриц.

Исследование производительности:







Для столбцов:

Размер матрицы	Кол-во процессов			
	1	4	9	16
100	0.018 мс	0.005 мс	0.003 мс	0.002 мс
500	0.472 мс	0.122 мс	0.079 мс	0.062 мс
1000	1.948 мс	0.605 мс	0.423 мс	0.394 мс
2500	12.219 мс	3.445 мс	2.449 мс	2.855 мс

Для строк:

Размер матрицы	Кол-во процессов			
	1	4	9	16
100	0.003 мс	0.003 мс	0.005 мс	0.018 мс
500	0.470 мс	0.116 мс	0.074 мс	0.053 мс
1000	1.852 мс	0.466 мс	0.260 мс	0.217 мс
2500	11.993 мс	2.993 мс	1.636 мс	1.326 мс

Для блоков:

Размер матрицы	Кол-во процессов			
	1	4	9	16
100	0.019 мс	0.005 мс	0.003 мс	0.002 мс
500	0.471 мс	0.120 мс	0.079 мс	0.068 мс
1000	1.939 мс	0.520 мс	0.304 мс	0.250 мс
2500	12.346 мс	3.322 мс	1.991 мс	1.868 мс

Для столбцов:

Размер матрицы 100:
При 4 потоках: ускорение ≈ 3.60, эффективность ≈ 0.90
При 9 потоках: ускорение ≈6.00≈6.00, эффективность ≈0.67≈0.67
При 16 потоках: ускорение ≈9.00≈9.00, эффективность ≈0.56≈0.56

Размер матрицы 500:
При 4 потоках: ускорение ≈ 3.87, эффективность ≈ 0.97
При 9 потоках: ускорение ≈ 5.97, эффективность ≈ 0.66
При 16 потоках: ускорение ≈ 7.61, эффективность ≈ 0.48

Размер матрицы 1000:
При 4 потоках: ускорение ≈ 3.22, эффективность ≈ 0.81
При 9 потоках: ускорение ≈ 4.60, эффективность ≈ 0.51
При 16 потоках: ускорение ≈ 4.95, эффективность ≈ 0.31

Размер матрицы 2500:
При 4 потоках: ускорение ≈ 3.55, эффективность ≈ 0.89
При 9 потоках: ускорение ≈ 4.99, эффективность ≈ 0.55
При 16 потоках: ускорение ≈ 4.28, эффективность ≈ 0.27

Для строк:

Размер матрицы 100:
При 4 потоках: ускорение ≈ 1.00, эффективность ≈ 0.25
При 9 потоках: ускорение ≈ 0.60, эффективность ≈ 0.07
При 16 потоках: ускорение ≈ 0.17, эффективность ≈ 0.01

Размер матрицы 500:
При 4 потоках: ускорение ≈ 4.05, эффективность ≈ 1.01
При 9 потоках: ускорение ≈ 6.35, эффективность ≈ 0.71
При 16 потоках: ускорение ≈ 8.87, эффективность ≈ 0.55

Размер матрицы 1000:
При 4 потоках: ускорение ≈ 3.97, эффективность ≈ 0.99
При 9 потоках: ускорение ≈ 7.12, эффективность ≈ 0.79
При 16 потоках: ускорение ≈ 8.52, эффективность ≈ 0.53

Размер матрицы 2500:
При 4 потоках: ускорение ≈ 4.01, эффективность ≈ 1.00
При 9 потоках: ускорение ≈ 7.33, эффективность ≈ 0.81
При 16 потоках: ускорение ≈ 9.02, эффективность ≈ 0.56

Для блоков:

Размер матрицы 100:
При 4 потоках: ускорение ≈ 3.80, эффективность ≈ 0.95
При 9 потоках: ускорение ≈ 6.33, эффективность ≈ 0.70
При 16 потоках: ускорение ≈ 9.50, эффективность ≈ 0.59

Размер матрицы 500:
При 4 потоках: ускорение ≈ 3.93, эффективность ≈ 0.98
При 9 потоках: ускорение ≈ 5.96, эффективность ≈ 0.66
При 16 потоках: ускорение ≈ 6.93, эффективность ≈ 0.43

Размер матрицы 1000:
При 4 потоках: ускорение ≈ 3.73, эффективность ≈ 0.93
При 9 потоках: ускорение ≈ 6.38, эффективность ≈ 0.71
При 16 потоках: ускорение ≈ 7.74, эффективность ≈ 0.48

Размер матрицы 2500:
При 4 потоках: ускорение ≈ 3.72, эффективность ≈ 0.93
При 9 потоках: ускорение ≈ 6.20, эффективность ≈ 0.69
При 16 потоках: ускорение ≈ 6.61, эффективность ≈ 0.41

Эти расчеты показывают, что с увеличением размера матрицы эффективность использования большего числа процессов увеличивается, что свидетельствует о лучшей масштабируемости алгоритма на больших данных. Однако, при этом наблюдается снижение эффективности с увеличением числа процессов, что указывает на наличие накладных расходов и затрат на коммуникацию между процессами. Это особенно заметно для меньших матриц и большего числа процессов.

2. Умножение матриц с разбиением на блоки по алгоритму Кэннона

Краткое описание функций и алгоритмов:

fillArrays: Инициализирует матрицы A и B фиксированными значениями и обнуляет матрицу C.

multiplyMatrixBlock: Выполняет блочное умножение матриц. Эта функция реализует стандартное умножение матриц, но на блоках матрицы.

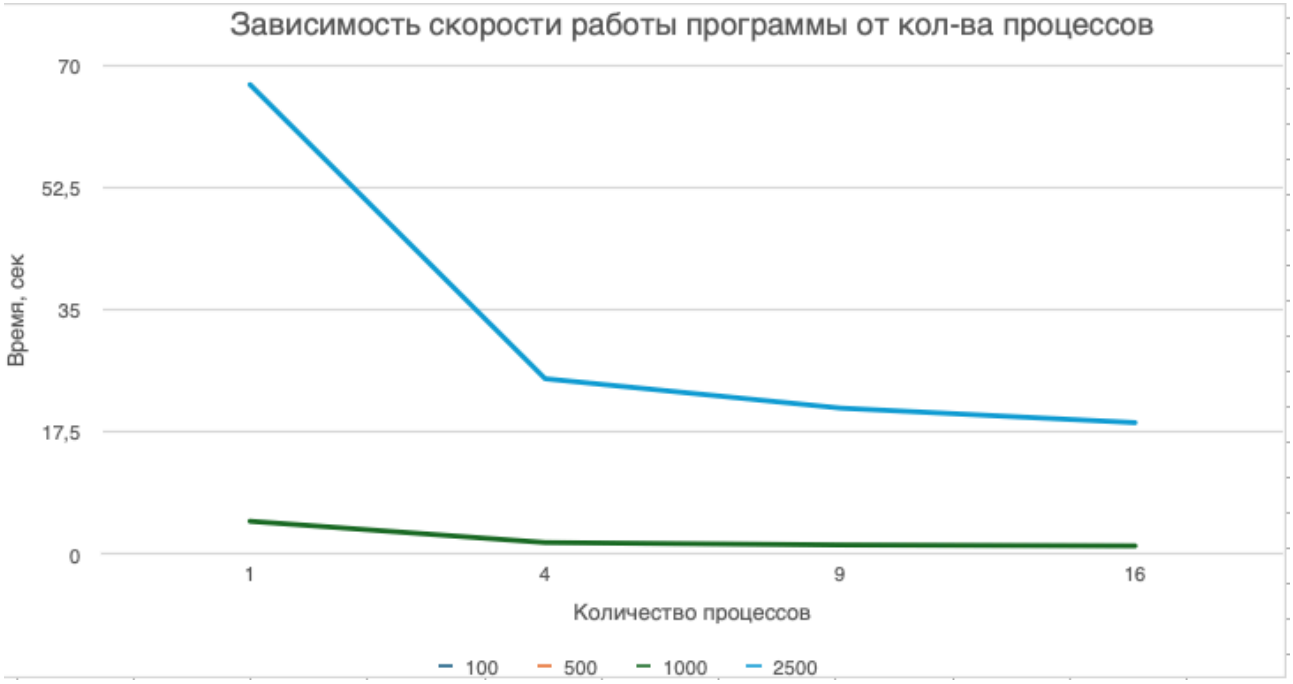
main: Главная функция, которая инициализирует MPI (Message Passing Interface), создает двумерную топологию для процессов (сетку), и управляет распределением данных и вычислительными процессами. Основная логика включает в себя инициализацию матриц, распределение блоков матриц между процессами, выполнение последовательных шагов умножения и пересылки блоков данных между процессами (согласно алгоритму Кэннона), и сбор результатов.

Формат ввода/вывода:

Ввод: Размер матрицы задается в коде (в данном случае, N = 2500), её содержимое генерируется случайно, количество процессов определяется через MPI.

Вывод: Время выполнения операции умножения матриц.

Исследование производительности:



Размер матрицы	Кол-во ghjwtccjd			
	1	4	9	16
100	0.005233	0.0023	0.0020	0.0069
500	0.5844	0.1878	0.1526	0.1331
1000	4,7038	1,6514	1,3081	1,1729
2500	67,2875	25,1448	20,9314	18,8386

Эта таблица представляет данные замеров времени работы программы, использующей технологию MPI для параллельных вычислений. В ней представлены разные размеры матриц (100, 500, 1000 и 2500) в строках и количество используемых процессов MPI (1, 4, 9, 16) в столбцах.

Данные получены при замерах на оборудовании со следующими вычислительными характеристиками:
• CPU: 4-ядерный AMD Ryzen 5 3550H 2,1–3,7 ГГц
• RAM: 8 ГБ DDR4

Из таблицы мы видим следующее:
• Размер матрицы 100: при увеличении количества процессов с 1 до 16 время уменьшается, но изменения несущественные, что может быть связано с низкой вычислительной сложностью задачи для такого размера матрицы.
• Размер матрицы 500: значительное уменьшение времени при увеличении количества процессов. Это указывает на эффективное распараллеливание задачи.
• Размер матрицы 1000: аналогично предыдущему, наблюдается значительное уменьшение времени выполнения при увеличении числа процессов.
• Размер матрицы 2500: наибольшее уменьшение времени выполнения. Это подтверждает, что MPI эффективнее всего работает на задачах с высокой вычислительной сложностью.

Чтобы сделать исследование производительности по данным из таблицы, нужно провести анализ представленных данных. Можно вычислить ускорение и эффективность для каждого случая. Ускорение S определяется как отношение времени работы на одном процессе к времени работы на n процессах. Эффективность E вычисляется как отношение ускорения к количеству процессов n: $E=S/n$.

Для матрицы размером 100:
При 4 потоках: ускорение составило примерно 2.28, а эффективность — 0.57.

При 9 потоках: ускорение было около 2.62, эффективность — 0.29.
При 16 потоках: ускорение составило около 0.76, что указывает на замедление по сравнению с однопоточным выполнением, а эффективность была около 0.05.

Для матрицы размером 500:
При 4 потоках: ускорение — около 3.11, эффективность — 0.78.
При 9 потоках: ускорение — около 3.83, эффективность — 0.43.
При 16 потоках: ускорение — около 4.39, эффективность — 0.27.

Для матрицы размером 1000:
При 4 потоках: ускорение — около 2.85, эффективность — 0.71.
При 9 потоках: ускорение — около 3.60, эффективность — 0.40.
При 16 потоках: ускорение — около 4.01, эффективность — 0.25.

Для матрицы размером 2500:
При 4 потоках: ускорение — около 2.68, эффективность — 0.67.
При 9 потоках: ускорение — около 3.21, эффективность — 0.36.
При 16 потоках: ускорение — около 3.57, эффективность — 0.22.