****МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

**Практикум по курсу**

**"Суперкомпьютеры и параллельная обработка данных"**

**Разработка параллельной версии программы для транспонирования матриц**

**ОТЧЕТ**

**о выполненном задании**

студентки 328 учебной группы факультета ВМК МГУ

Шелепнёвой Дарьи Дмитриевны

Москва, 2021 г.

Оглавление

[1 Постановка задачи - 2 -](#_Toc90236953)

[2 Описание алгоритма транспонирования матрицы - 2 -](#_Toc90236954)

[2.1 Основа: последовательный алгоритм - 2 -](#_Toc90236955)

[2.2 Параллельный алгоритм - 3 -](#_Toc90236956)

[3 Результаты замеров времени выполнения - 4 -](#_Toc90236957)

[3.1 Таблицы - 4 -](#_Toc90236958)

[3.2 Графики - 5 -](#_Toc90236959)

[4 Анализ результатов - 7 -](#_Toc90236960)

[5 Выводы - 7 -](#_Toc90236961)

# Постановка задачи

Ставится задача транспонирования матрицы.

Дана матрица , требуется получить матрицу , где .

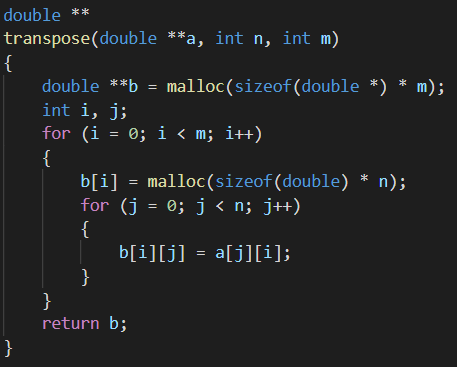
Требуется:

1. Реализовать параллельный алгоритм транспонирования матрицы с помощью технологий параллельного программирования OpenMP и MPI.
2. Сравнить их эффективность.
3. Исследовать масштабируемость полученных программ и построить графики зависимости времени выполнения программ от числа используемых потоков и объёма входных данных.

# Описание алгоритма транспонирования матрицы

## Основа: последовательный алгоритм

Простейший алгоритм транспонирования матрицы имеет следующий вид:

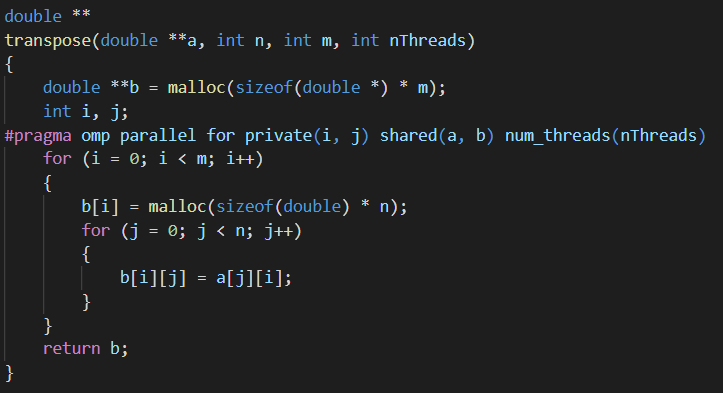


Этот алгоритм имеет сложность .

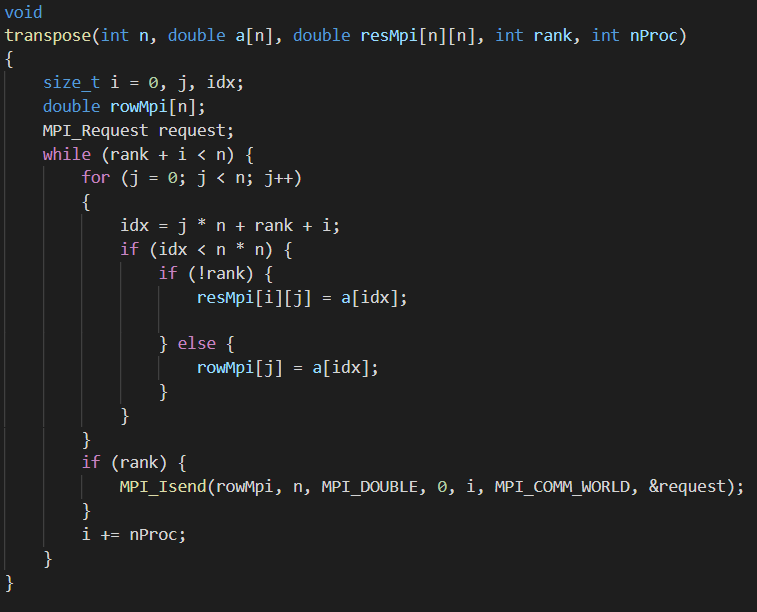
## Параллельный алгоритм

Разбиваем задачу вычисления конечной матрицы на подзадачи по вычислению строк и распределяем их по потокам. Разбиение на вычисление отдельных полей не производим, т.к. размеры матриц при вычислениях и так будут на порядки превышать число потоков/процессов.

В OpenMP модификация кода сводится к добавлению ***omp parallel for.***



В MPI-версии производится широковещательная рассылка заполненной матрицы ***a***, каждый процесс изменяет соответствующие строки и отправляет их процессу-мастеру (имеющий rank=0) с помощью команд ***MPI\_Isend*** и ***MPI\_Irecv***. Для синхронизации используются команды ***MPI\_Barrier.***



Коды программ можно посмотреть в репозитории <https://github.com/DariaShel/skipod> в соответствующих папках.

# Результаты замеров времени выполнения

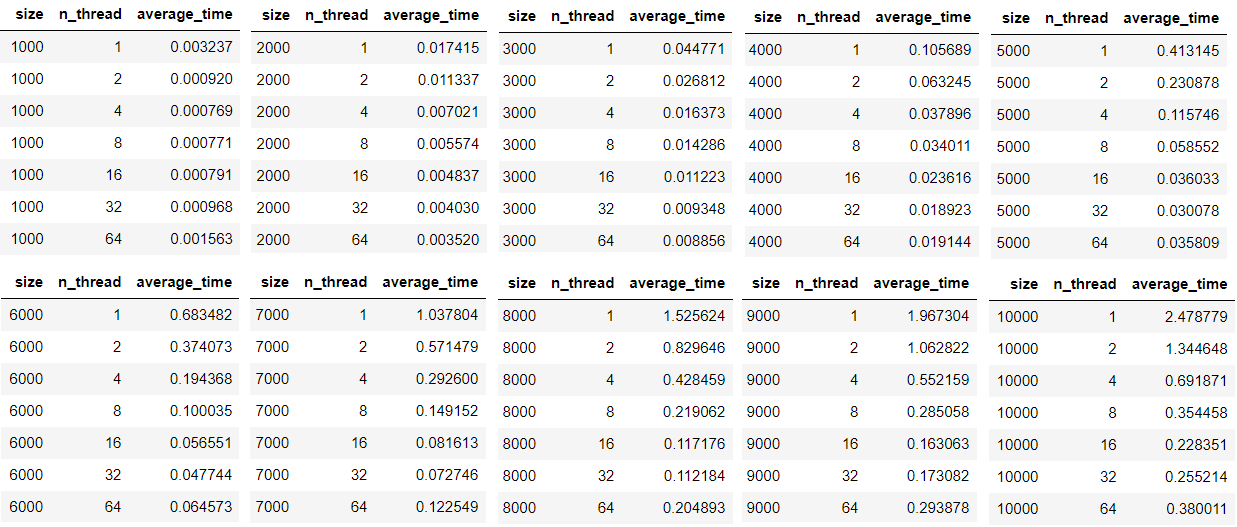
Ниже приведены результаты замеров времени выполнения программ на суперкомпьютере Polus в табличной форме и наглядно на графиках.

Программы запускались со следующими параметрами:

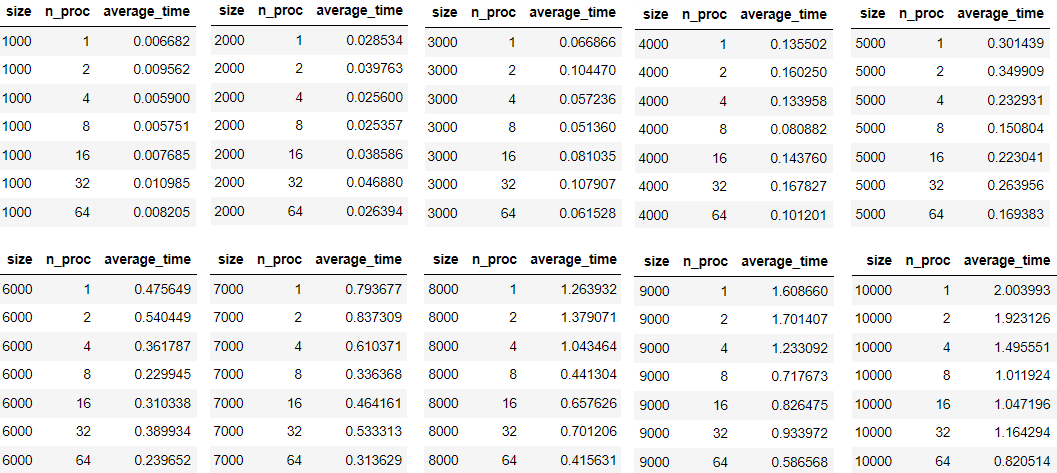
* Матрица
* Количество потоков (процессов)

Было взято среднее значение времени за 5 запусков для каждой конфигурации.

## Таблицы



Результаты алгоритма OpenMP

**

Результаты алгоритма MPI

## Графики

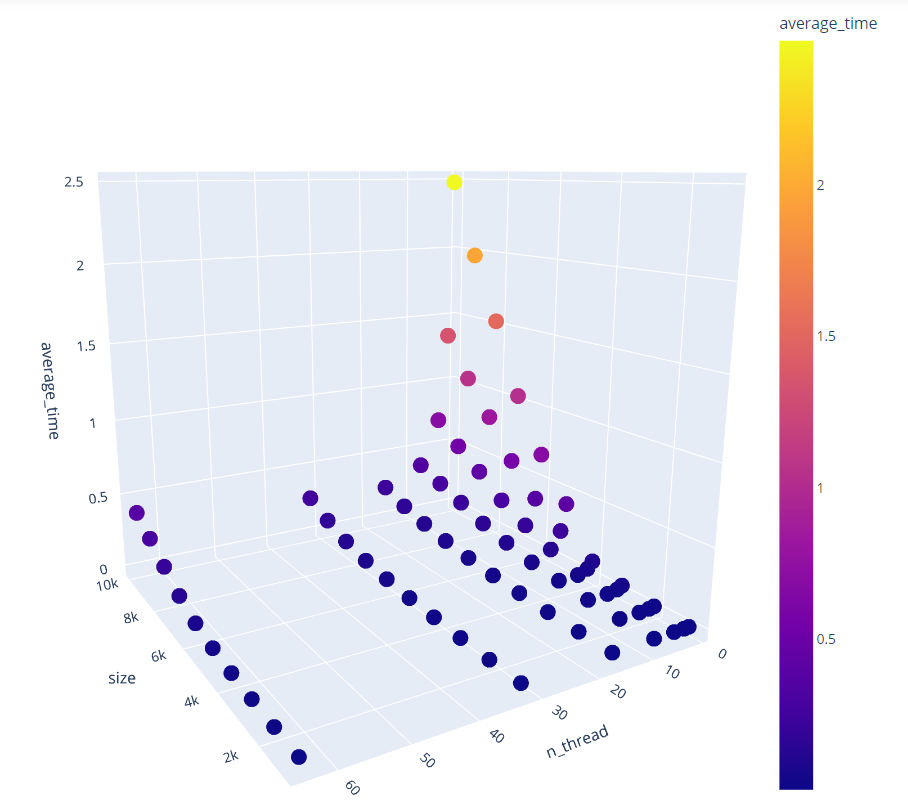


График1 (OpenMP)

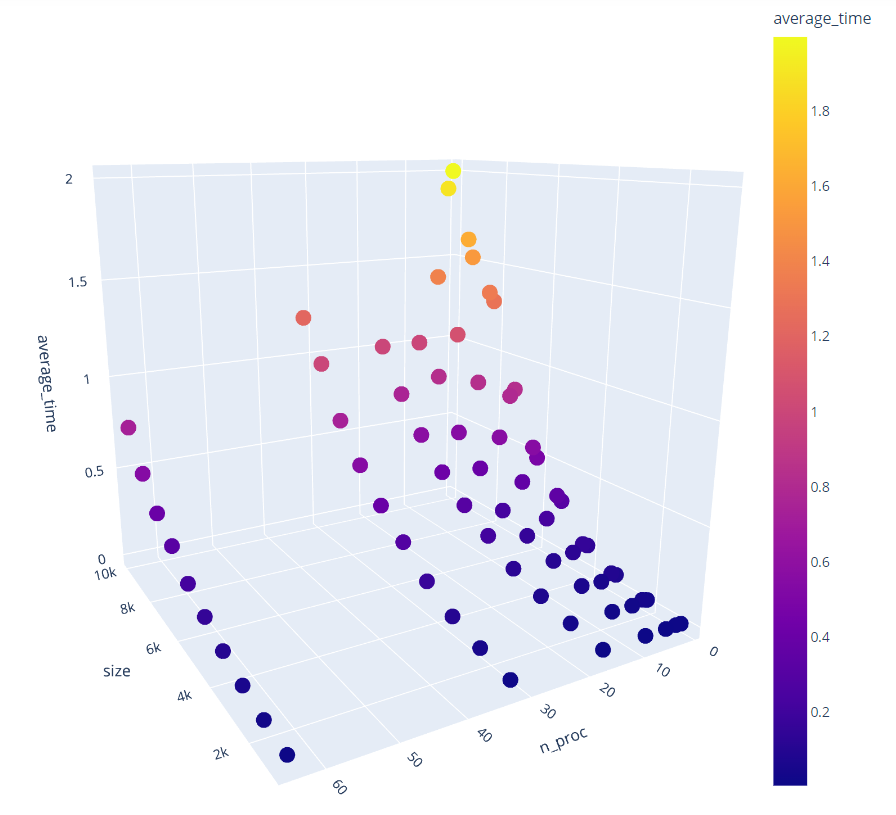


График1 (MPI)

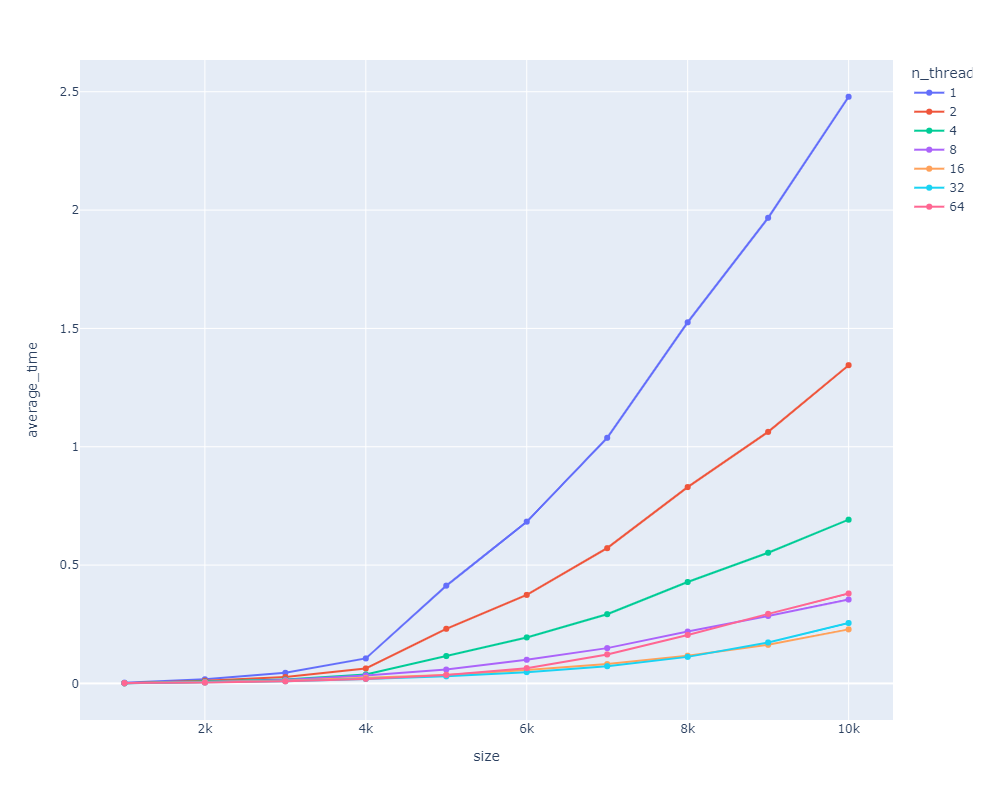


График2 (OpenMP)

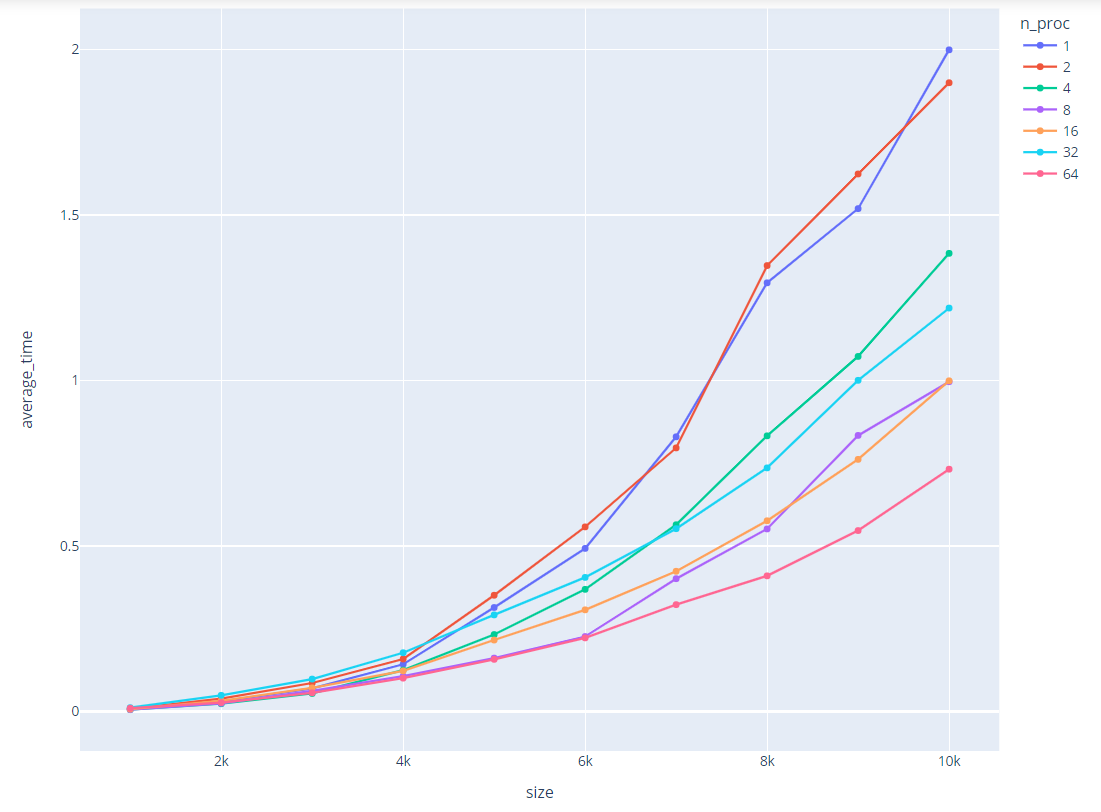
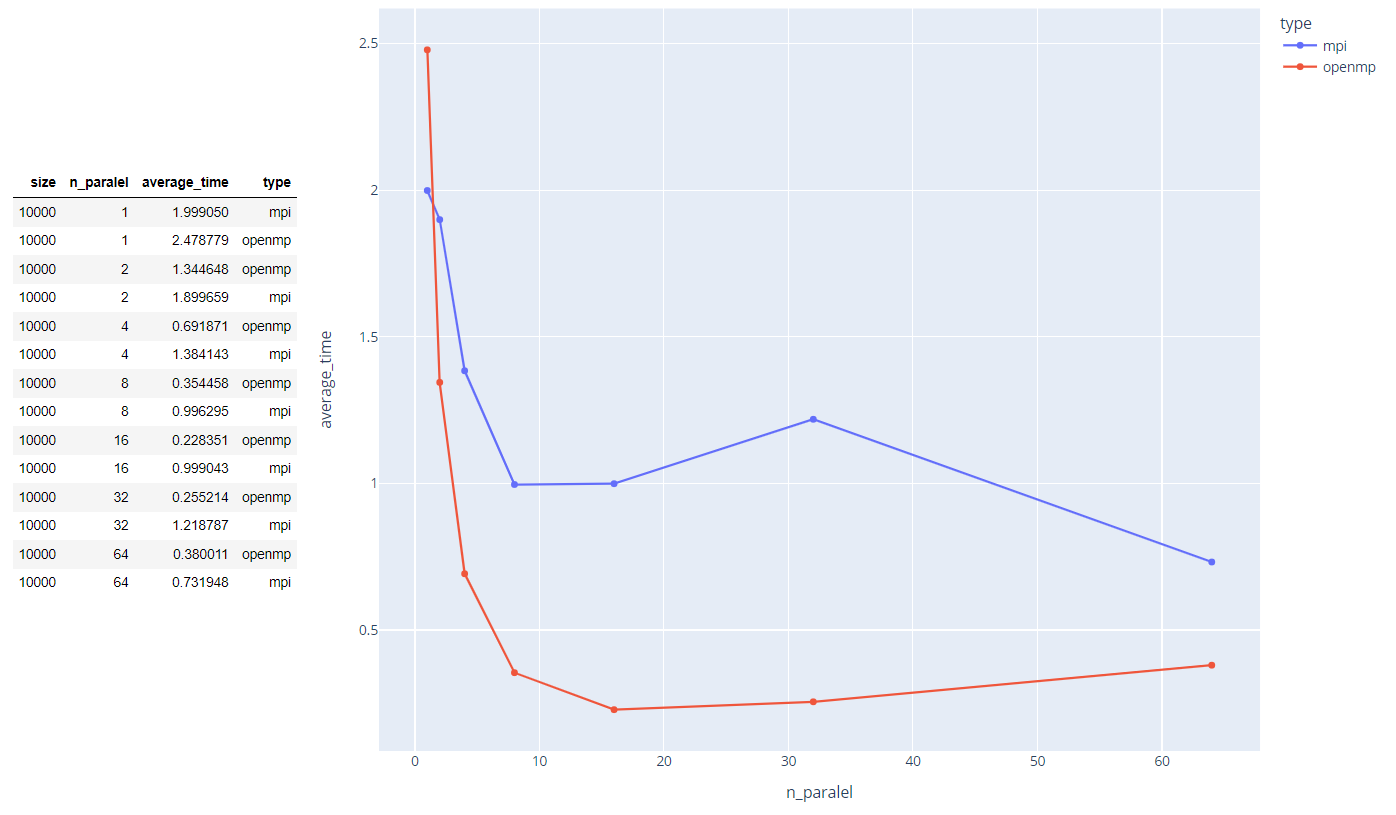


График2 (MPI)

# Анализ результатов

На одинаковых конфигурациях OpenMP показал результаты лучшие, чем MPI. В алгоритме MPI тратится дополнительное время на пересылку данных между процессорами. Для наглядности был построен график зависимости скорости выполнения алгоритма от количества потоков/процессов для матрицы размера .



Сравнение алгоритмов OpenMP и MPI

Как видно из графика, OpenMP справляется с поставленной задачей гораздо быстрее, чем MPI.

# Выводы

Выполнена работа по разработке параллельной версии алгоритма транспонирования матриц. Изучена технология написания параллельных алгоритмов OpenMP и MPI. Проанализировано время выполнения алгоритмов на вычислительной системе Polus.

Технология OpenMP достаточно удобна в использовании и даёт значительный прирост производительности на рассчитанных на многопоточные вычисления системах.

MPI можно назвать более низкоуровневой технологией: разработка MPI-программы знакомит с основами взаимодействия вычислительных узлов суперкомпьютера.