

ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені ІВАНА ФРАНКА
Факультет прикладної математики та інформатики

Паралельні та розподілені обчислення
ЛАБОРАТОРНА РОБОТА №9-10

Виконала:
Ст. Пелещак Вероніка
ПМІ-35с

Тема: Паралелізація методу множення матриць засобами MPI

Постановка задачі: Реалізувати множення матриць: A — розміром $n \times m$, B — розміром $m \times r$, отримати матрицю C розміром $n \times r$.

Реалізувати два варіанти:

- 1. Послідовний варіант** множення.
- 2. Паралельний варіант** з використанням MPI:
 - розподіл рядків A між процесами,
 - ширококомовна передача B ,
 - паралельне обчислення частин C ,
 - збір результатів у кореневому процесі.

Виконати MPI-програми множення матриць на обчислювальному кластері.

Опис MPI-алгоритму

1. Ініціалізація MPI:

`MPI_Init()`

`MPI_Comm_rank()`

`MPI_Comm_size()`

2. Генерація матриць у процесі 0:

Процес 0 створює матриці A та B , заповнює їх випадковими значеннями та виконує послідовне множення для подальшого порівняння.

3. Розсилка розмірів матриць всім процесам:

`MPI_Bcast(&n)`

`MPI_Bcast(&m)`

`MPI_Bcast(&p)`

Всі процеси повинні знати розміри матриць.

4. Передача всієї матриці В усім процесам:

`MPI_Bcast(B)`

Матриця В однакова для всіх процесів, тому її передаємо повністю.

5. Розподіл матриці А між процесами:

З використанням `MPI_Scatterv`:

- `sendCounts[i]` — скільки елементів передати процесу *i*
- `displs[i]` — з якого індекса брати
- формула для нерівномірного розподілу (коли $n \% size \neq 0$)

6. Паралельне множення:

Кожен процес множить свої рядки А на всю матрицю В.

for *i* in `myRows`:

for *j* in `p`:

for *k* in `m`:

`C_local[i][j] += A_local[i][k] * B[k][j]`

7. Збір результатів у процес 0:

`MPI_Gatherv()`

8. Порівняння з послідовним варіантом:

Перевірка правильності:

`if (C == Cseq) — Ок`

9. Обчислення прискорення і ефективності:

$\text{speedup} = T_{\text{seq}} / T_{\text{par}}$

$\text{efficiency} = \text{speedup} / \text{size}$

Структура програми

Функції:

- `fillMatrix()` — генерація матриці
- `multiplySequential()` — послідовне множення
- MPI-блок — паралельна частина

Основна логіка:

- ініціалізація MPI
- розподіл даних
- паралельне множення
- збір результатів
- вимірювання часу

Результати роботи програми:

1. Зі значенням за замовчуванням(1000x1000):

```
veronikapelesak@Noutbuk-Veronika cmake-build-debug % mpirun -np 6 ./matrix_mpi
Sequential time: 1644.466 ms
Parallel (MPI) time: 308.238 ms
Sequential (re-measured) time: 1624.234 ms
Speedup: 5.269
Efficiency: 0.878
Result check: OK
veronikapelesak@Noutbuk-Veronika cmake-build-debug %
```

2. Розмірність кратна кількості процесів:

```
veronikapelesak@Noutbuk-Veronika cmake-build-debug % mpirun -np 4 ./matrix_mpi 400 400 400
Sequential time: 100.714 ms
Parallel (MPI) time: 25.236 ms
Sequential (re-measured) time: 99.828 ms
Speedup: 3.956
Efficiency: 0.989
Result check: OK (C == Cseq)
veronikapelesak@Noutbuk-Veronika cmake-build-debug %
```

```
veronikapelesak@Noutbuk-Veronika cmake-build-debug % mpirun -np 5 ./matrix_mpi 1000 500 300
Sequential time: 239.155 ms
Parallel (MPI) time: 47.918 ms
Sequential (re-measured) time: 235.309 ms
Speedup: 4.911
Efficiency: 0.982
Result check: OK
veronikapelesak@Noutbuk-Veronika cmake-build-debug %
```

3. Розмірність НЕ кратна кількості процесів:

```
veronikapelesak@Noutbuk-Veronika cmake-build-debug % mpirun -np 4 ./matrix_mpi 1025 1025 1025
Sequential time: 1742.466 ms
Parallel (MPI) time: 507.607 ms
Sequential (re-measured) time: 1754.278 ms
Speedup: 3.456
Efficiency: 0.864
Result check: OK
veronikapelesak@Noutbuk-Veronika cmake-build-debug %
```

Компіляція та запуск на кластері:

1. Створення каталогу: *mkdir peleshchak_lab10*.
2. Завантаження файлів: *через scp*.
3. Компіляція: *mpicc -std=gnu99 matrix_mpi.c -o matrix_mpi*.
4. Створення файлу *mpi.qsub*:

```
#!/bin/bash
```

```
#$ -N matrix_mpi_job
```

```
#$ -cwd
```

```
#$ -j y
```

```
echo "Running on host: $(hostname)"
```

```
echo "Using N slots: $NSLOTS"
```

```
mpirun -np $NSLOTS ./matrix_mpi 1000 1000 1000
```

5. Запуск задачі на кластері: *qsub -pe mpi 2/4/6/8/12 mpi.qsub*.

Результати:

job-ID	prior	name	user	state	submit/start at	queue	slots	ja-task-ID
1518	0.52246	matrix_mpi	user11	qw	11/30/2025 00:24:00		12	
1517	0.51611	matrix_mpi	user11	qw	11/30/2025 00:22:47		8	
1516	0.51294	matrix_mpi	user11	qw	11/30/2025 00:22:32		6	
1519	0.50976	matrix_mpi	user11	qw	11/30/2025 00:24:57		4	
1509	0.50659	matrix_mpi	user11	qw	11/29/2025 23:09:57		2	

[user11@cluster-edu peleshchak_lab10]\$

Висновок:

У ході виконання лабораторної роботи №9–10 було реалізовано паралельний алгоритм множення матриць із використанням MPI та виконано розподіл обчислень між процесами. Програма коректно працює для різної кількості процесів, включно з випадками, коли розмір матриці не ділиться порівну.

Було здійснено компіляцію та підготовку запуску на кластері. Завдання успішно подавалися в чергу, що підтверджує правильність налаштування файлу *mpi.qsub*. Однак через невідому для мене причину всі задачі залишилися у статусі *qw*, тому фактичний запуск на вузлах не відбувся.