НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ім. І. СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра автоматики та управління в технічних системах

Лабораторна робота №7

із дисципліни «Технології паралельних та розподілених обчислень»

на тему «Розробка паралельного алгоритму множення матриць з використанням MPI-методів колективного обміну повідомленнями («один-до-багатьох», «багато-до-одного», «багато-до-багатьох») та дослідження його ефективності»

Підготував:

Студент ФІОТ, гр. ІТ-83,

Троян О.С,

Перевірив:

пос. Дифучина О. Ю.

Київ 2020

Мета роботи: Ознайомитись з методами колективного обміну повідомленнями типу «один-до-багатьох», «багато-до-одного», «багато-до-багатьох» (див. лекцію та документацію стандарту MPI).
Завдання:

1. Реалізувати алгоритм паралельного множення матриць з використанням розподілених обчислень в MPI з використанням методів колективного обміну повідомленнями.
   40 балів.
2. Дослідити ефективність розподіленого обчислення алгоритму множення матриць при збільшенні розміру матриць та при збільшенні кількості вузлів, на яких здійснюється запуск програми. Порівняйте ефективність алгоритму при використанні методів обміну повідомленнями «один-до-одного», «один-до-багатьох», «багато-до-одного», «багато-до-багатьох».
   60 балів.

Хід роботи

1. Один до багатьох

```
var rowsPerNode:int = rank / comm.Size;

var block:int[][] = comm.ScatterFromFlattened(first, count:rowsPerNode, root:Master);
comm.Broadcast(ref second, root:Master);

var resultBlock:int[][] = Multiply(block, second);

comm.Gather(resultBlock, root:Master);

if (comm.Rank != Master) return;
if (stopwatch == null) return;

stopwatch.Stop();
Console.WriteLine("Multiplication time elapsed - " + stopwatch.ElapsedMilliseconds);
Console.WriteLine("Time elapsed - " + (initializationTime + stopwatch.ElapsedMilliseconds));
```

За допомогою Broadcast та Scatter розсилаємо повідомлення всім процесам, а потім за допомогою Gather що є зв'язком всіх до одного, надсилаємо результати обчислень до Master процесу.

Результат:

```
Initialization time elapsed - 1
Multiplication time elapsed - 171
Time elapsed - 172
```

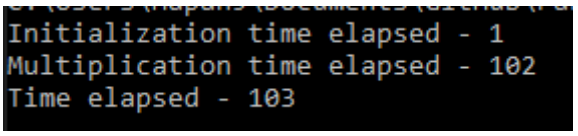
2. Багато до багатьох

```
var second:int[][] = comm.AllgatherFlattened(secondBlock, secondBlock.Length);

var resultBlock:int[][] = Multiply(firstBlock, second);

comm.Gather(resultBlock, root:Master);
```

За допомогою Allgather збираємо дані з всіх процесів і зберігаємо в буфері всіх процесів.

Після розрахунків передаємо данні до Master.

Результат:



3. Дослідимо швидкодію один до багатьох и багато до багатьох при різній кількості процесів та розмірності матриць.

Ранг	9	27	81	81	600	600
Процеси	3	3	9	3	3	6
OtM	164	159	1487	198	3226	3024
MtM	119	171	319	154	1879	1405

Висновок: під час виконання даної лабораторної роботи я розробив паралельний алгоритм множення матриць з використанням MPI-методів колективного обміну повідомленнями («один-до-багатьох» та «багато-до-багатьох») та дослідив його ефективність.