**Лабораторная работа №3**

**Макеева Настя ПИН-31**

**Обобщенная передача данных**

**Цель:** изучить основные принципы обобщенной передачи данных в технологии MPI на примере использования в рамках языка С++.

**Лабораторные задания**

**Задание.** Модифицировать программу, написанную на Л.Р. №1 так чтобы она работала на основе обобщенной передачи сообщений. **Результаты работы сравнить (с результатами полученными в Л.Р. №2) и занести в отчет.**

При обобщенной операции передачи данных от одного процесса всем процессам (при распределении данных) передача данных происходит быстрее всего. Scatter отличается от широковещательной рассылки тем, что процесс передает процессам различающиеся данные.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **4** | **10** | **100** |
| **Lab 2 (BCast)** | **0.002578** | **0.003391** | **0.010978** |
| **Lab3 (Scatter)** | **0.000496** | **0.000591** | **0.000621** |

Лаба2 (добавлены замеры времени)

#include <stdio.h>

#include <mpi.h>

#include <iostream>

int main(int argc, char\* argv[])

{

int M = 100;

int ProcRank, ProcNum;

int a = 0;

int mes = 0;

double t1, t2, time;

MPI\_Init(&argc, &argv);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &ProcNum);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &ProcRank);

if (ProcRank == a)

t1 = MPI\_Wtime();

for (int i = 0; i < M \* ProcNum; i++)

{

//if (a == ProcNum) a = 1;

a = (a + 1) % ProcNum;

if (ProcRank == a)

mes += ProcRank;

MPI\_Bcast(&mes, 1, MPI\_INT, a, MPI\_COMM\_WORLD);

if (ProcRank == a)

printf("Proc mumber: %d send message: %d from proc number %d for the %d time\n", ProcRank, mes, (a) % ProcNum, i);

if (ProcRank == (a + 1) % ProcNum)

printf("Proc mumber: %d recieve message: %d for the %d time\n", ProcRank, mes, i);

}

//MPI\_Bcast(&message, 1, MPI\_CHAR, 2, MPI\_COMM\_WORLD);

//printf("All processes received message : \"%s\" from processor number %d for the %d time \n\n", /\*ProcRank,\*/ //message, recvFrom, i);

if (ProcRank == a)

{

t2 = MPI\_Wtime();

time = t2 - t1;

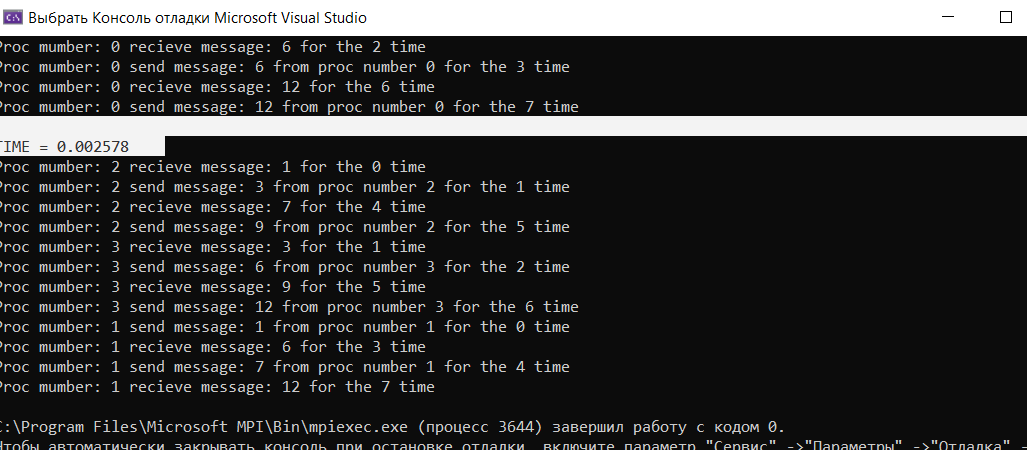
printf("\nTIME = %f\n", time);

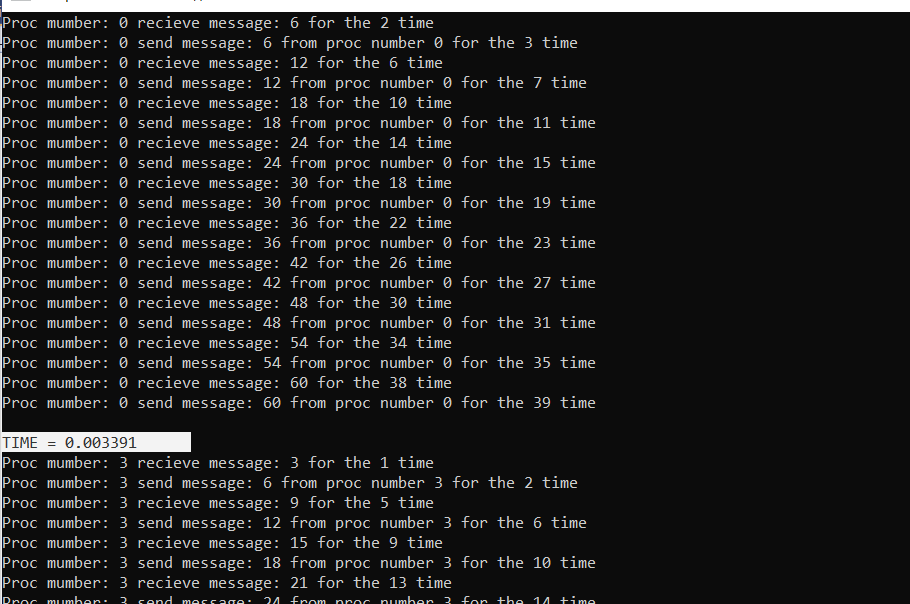
}

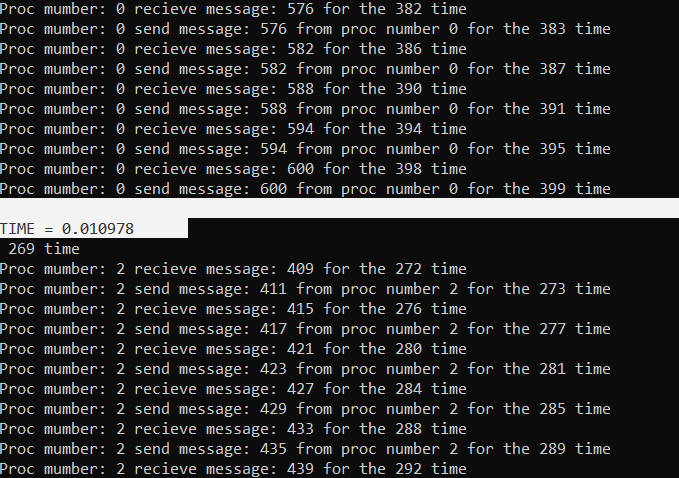
MPI\_Finalize();

return 0;

}







Лаб 3

#include <stdio.h>

#include <mpi.h>

#include <iostream>

int main(int argc, char\* argv[])

{

int M = 100;

int ProcRank, ProcNum;

int a = 0;

int mes = 0;

MPI\_Status Status;

int\* mesMain = nullptr;

double t1, t2, time;

MPI\_Init(&argc, &argv);

MPI\_Comm\_size(MPI\_COMM\_WORLD, &ProcNum);

MPI\_Comm\_rank(MPI\_COMM\_WORLD, &ProcRank);

if (ProcRank == a)

t1 = MPI\_Wtime();

mesMain = new int[ProcNum];

for (int i = 0; i < M; i++)

{

for (int i = 0; i < ProcNum; i++)

{

if (ProcRank == a)

mesMain[i] = 1;

else

mesMain[i] = 0;

}

//a = (a + 1) % ProcNum;

if (ProcRank == a)

mes += ProcRank;

// отправить всем

MPI\_Scatter(mesMain, 1, MPI\_INT, &mes, 1, MPI\_INT, 0, MPI\_COMM\_WORLD);

//MPI\_Bcast(&mes, 1, MPI\_INT, a, MPI\_COMM\_WORLD);

if (ProcRank != a && mes == 1) {

// printf("Proc mumber: %d send message: %d from proc number %d for the %d time\n", ProcRank, mes, (a) % ProcNum, i);

//if (ProcRank == (a + 1) % ProcNum)

printf("Proc mumber: %d recieve message: %d for the %d time\n", ProcRank, mes, i);

MPI\_Gather(&mes, 1, MPI\_INT, mesMain, 1, MPI\_INT, 0, MPI\_COMM\_WORLD);

}

}

if (ProcRank == a)

{

t2 = MPI\_Wtime();

time = t2 - t1;

printf("\nTIME = %f\n", time);

}

MPI\_Finalize();

return 0;

}

