**Практическая работа №1**

**Вариант №5**

**Формулировка задания.**

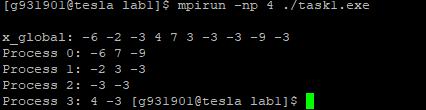
На процессе с номером ноль заполнить массив x\_global[] случайными числами из диапазона от -9 до 9. Размер массива задать равным n. Распределить вектор x\_global[] по процессам раздавая на каждый по одному элементу.

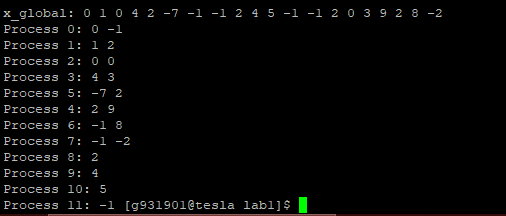
**Ход работы.**

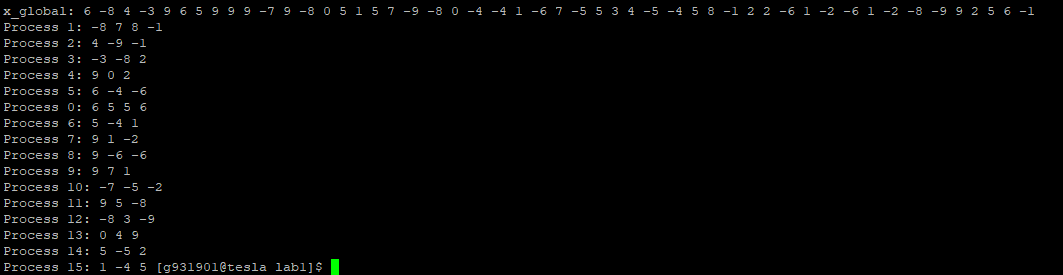
Схема вычислений:

Сначала мы формируем x\_global[n]. После чего определяем длину массива x\_local для каждого процесса, которая определяется n / size и, в случае, если у нас есть остаток от деления size на n мы добавляем по одному элементу только тем процессам, которые меньше этого остатка. Потом мы отправляем каждому процессу в цикле по одному элементу. В конце мы принимаем на каждом процессе отправленные элементы и выстраиваем x\_local.

Пример(size = 4, 12, 16 и n = 10, 20, 50):







**Заключение**

В данной работе были изучены двухточечные операции в MPI. Была написана программа для отправки данных из одного процееса в другой, заполнением локального массива, с последующим выводом на экран.

**Код программы**

#include <stdio.h>

#include <mpi.h>

#include <cstdlib>

#include <ctime>

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

{

int rank, size, rc;

const int n = 10;

int x\_global[n];

MPI\_Status status;

rc = MPI\_Init(&argc, &argv);

rc = MPI\_Comm\_size(MPI\_COMM\_WORLD, &size);

rc = MPI\_Comm\_rank(MPI\_COMM\_WORLD, &rank);

int length;

if (rank < n % size) {

length = (n / size) + 1;

}

else {

length = n / size;

}

int x\_local[length];

if (rank == 0) {

srand(time(NULL));

for (int i = 0; i < n; i++) {

x\_global[i] = (rand() % 19) - 9;

}

printf("\nx\_global: ");

for (int i = 0; i < n; i++) {

printf("%d ", x\_global[i]);

}

for(int i = 1; i < size; i++){

if (i < n % size) {

for(int j = 0; j < length; j++) {

MPI\_Send(&x\_global[(j \* size) + i], 1, MPI\_INT, i, 0, MPI\_COMM\_WORLD);

}

}

else if(i >= n % size && n % size > 0) {

for(int j = 0; j < length - 1; j++) {

MPI\_Send(&x\_global[(j \* size) + i], 1, MPI\_INT, i, 0, MPI\_COMM\_WORLD);

}

}

else {

for(int j = 0; j < length; j++) {

MPI\_Send(&x\_global[(j \* size) + i], 1, MPI\_INT, i, 0, MPI\_COMM\_WORLD);

}

}

}

for(int i = 0; i < length; i++){

x\_local[i] = x\_global[i\*size];

}

}

else{

for(int i = 0; i < length; i++){

MPI\_Recv(&x\_local[i], 1, MPI\_INT, 0, 0, MPI\_COMM\_WORLD, &status);

}

}

printf("\nProcess %d: ", rank);

for (int i = 0; i < length; i++) {

printf("%d ", x\_local[i]);

}

rc = MPI\_Finalize();

}

**Ссылка на код на кластере**

/home/g931901/g932204/Sobol/lab1