по практическому заданию №3

Дисциплина: Распределенные задачи и алгоритмы

Выполнил:

Студент 45 группы

Худышкин Е. Д.

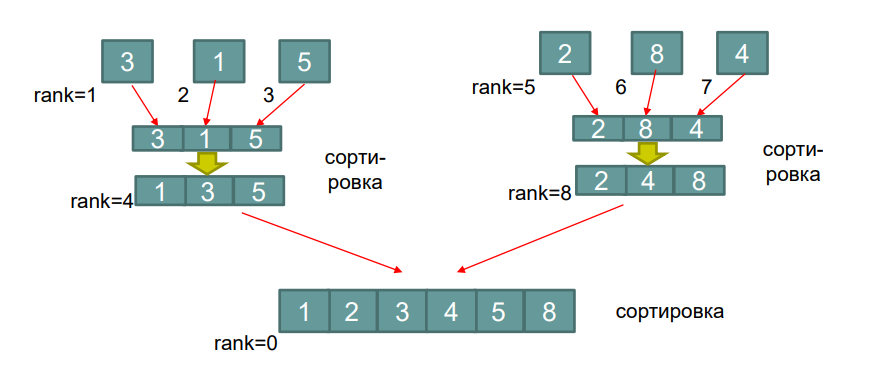
Преподаватель:

Шиян В. И.

**Постановка задачи:**

Реализовать так называемую задачу фильтрации,

используя неблокирующие обмены + Waitall(), например:



**Ход работы:**

В ходе работы было определено, что для правильной работы нам необходимо сделать так, чтобы поток №0 ожидал получения двух наборов чисел, от двух разных потоков, которые в свою очередь получают свои наборы по одному числу от каждого отдельного потока. Основываясь на данной концепции был создан массив из 6 чисел для сортировки, потоки под номерами 1, 2, 3 передают по одному числу из массива потоку №7, потоки под номерами 4, 5, 6 передают по одному оставшемуся числу из массива потоку №8; 7 и 8 потоки передают уже по 3 числа потоку №0, который всё это время ожидал получения используя Waitall(), поток сортирует и выводит полученный массив.

Текст программы:

**import mpi.\*;**

**import java.util.Arrays;**

**public class Main {**

**public static void main(String[] args) {**

**int[] numbers = { 1, 9, 4, 0, 4, -3 }; // сортируемый массив**

**MPI.Init(args);**

**int rank = MPI.COMM\_WORLD.Rank();**

**int size = MPI.COMM\_WORLD.Size();**

**if (rank == 0) {**

**Request[] requests = new Request[size - 1];**

**int[] finalBuf = new int[numbers.length];**

**for (int i = 0; i < 2; i++) {**

**requests[i] = MPI.COMM\_WORLD.Irecv(finalBuf, 3 \* i, 3, MPI.INT, 7 + i, 2);**

**}**

**Request.Waitall(requests); // Ожидаем получения**

**System.out.println("Unsorted massive:");**

**print(numbers);**

**System.out.println("Sorted massive:");**

**Arrays.sort(finalBuf);**

**print(finalBuf);**

**} else if (rank > 0 && rank < 7) {**

**int[] rBuf = { numbers[rank-1] };**

**int dest = rank < 4 ? 7 : 8;**

**MPI.COMM\_WORLD.Issend(rBuf, 0, 1, MPI.INT, dest, 1);**

**} else {**

**int[] rBuf = new int[3];**

**Request[] reqs = new Request[rBuf.length];**

**Status[] stats = new Status[reqs.length];**

**int n = rank == 7 ? 0 : 3;**

**for (int i = 1 + n; i < 4 + n; i++) {**

**int[] tBuf = new int[1];**

**reqs[(i - 1) % 3] = MPI.COMM\_WORLD.Irecv(rBuf, (i - 1) % 3, 1, MPI.INT, i, 1);**

**}**

**Request.Waitall(reqs);**

**Arrays.sort(rBuf);**

**MPI.COMM\_WORLD.Issend(rBuf, 0, 3, MPI.INT, 0, 2);**

**}**

**MPI.Finalize();**

**}**

**private static void print(int[] massive) {**

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

**System.out.print(massive[i]);**

**if (i < massive.length-1) {**

**System.out.print(", ");**

**} else System.out.println();**

**}**

**}**

**}**

Результаты работы программы при использовании 9 потоков (в соответствии с размером массива):

Изображение выглядит как текст

Автоматически созданное описание

Массив успешно отсортирован, программа отработала корректно.