

МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

ЛАБОРАТОРНАЯ РАБОТА №2
по дисциплине «Параллельные алгоритмы и системы»
Тема: Передача данных по процессам

Студент гр. 1307

Угрюмов М.М.

Преподаватель

Санкт-Петербург

2025

Введение

Тема работы: запуск параллельной программы и передача данных по процессам.

Цель работы: освоить функции передачи данных между процессами.

Лабораторная работа №2

Задание №1:

- 1) Запустить 4 процесса.
- 2) На каждом процессе создать переменные: a_i, b_i, c_i , где i – номер процесса. Инициализировать переменные. Вывести данные на печать.
- 3) Передать данные на другой процесс. Напечатать номера процессов и поступившие данные. Найти: $c_0 = a_1 + b_2$; $c_1 = a_3 + b_0$; $c_2 = a_0 + b_3$; $c_3 = a_2 + b_1$.

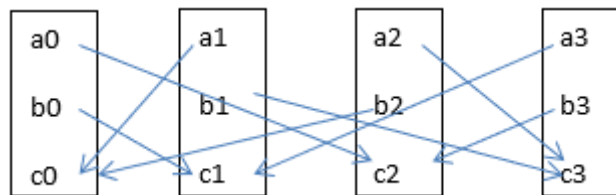


Рис.1 – Задание №1

Задание №2 (Вариант 6):

Запустить n процессов и найти количество четных положительных элементов.

Ход работы

Задание №1

Каждый процесс (0, 1, 2, 3) имеет заранее определенные отправляющие и принимающие процессы: `send_a_to` и `send_b_to` — массивы, которые содержат индексы процессов, на которые текущий процесс отправляет свои значения `a` и `b`. `recv_a_from` и `recv_b_from` — массивы, которые содержат индексы процессов, от которых текущий процесс будет получать значения `a` и `b`.

Каждый процесс вызывает метод `sendData(a, b, send_a_to, send_b_to)`, чтобы отправить свои значения переменных `a` и `b` на другие процессы.

```
MPI.COMM_WORLD.Send(new int[]{a}, 0, 1, MPI.INT, send_a_to, 99)
```

отправляет значение `a` на процесс, указанный в `send_a_to`.

```
MPI.COMM_WORLD.Send(new int[]{b}, 0, 1, MPI.INT, send_b_to, 99)
```

отправляет значение `b` на процесс, указанный в `send_b_to`.

После того как процесс отправил свои данные, он получает данные от других процессов, вызывая метод `receiveData(received_a, received_b, recv_a_from, recv_b_from)`.

```
MPI.COMM_WORLD.Recv(received_a, 0, 1, MPI.INT, recv_a_from, 99)
```

получает значение `a` от процесса, указанного в `recv_a_from`.

```
MPI.COMM_WORLD.Recv(received_b, 0, 1, MPI.INT, recv_b_from, 99)
```

получает значение `b` от процесса, указанного в `recv_b_from`.

После того как процесс получил данные от других процессов, он выводит полученные значения и вычисляет результат: сумму полученных значений `a` и `b`.

```

MPJ Express (0.44) is started in the multicore configuration
Process 0 (Thread ID: 22)
    Initial values: a0 = 1, b0 = 2
Process 2 (Thread ID: 24)
    Initial values: a2 = 3, b2 = 6
Process 1 (Thread ID: 25)
    Initial values: a1 = 2, b1 = 4
Process 3 (Thread ID: 23)
    Initial values: a3 = 4, b3 = 8

Process 3 received: a = 3, b = 4
    Computed value c3 = a + b = 3 + 4 = 7
Process 1 received: a = 4, b = 2
    Computed value c1 = a + b = 4 + 2 = 6
Process 2 received: a = 1, b = 8
    Computed value c2 = a + b = 1 + 8 = 9
Process 0 received: a = 2, b = 6
    Computed value c0 = a + b = 2 + 6 = 8

Process finished with exit code 0
|

```

Рис.2 – Результат выполнения задания №1

Задание №2

В качестве примера исходное кол-во процессов – $-np = 4$.

Каждый процесс получает часть общего массива из 10 случайных чисел, сгенерированного процессом 0, в диапазоне от -100 до 100. Каждый процесс вызывает метод `countPositive(localNumbers)`, который считает количество четных положительных чисел в полученной части массива. Каждый процесс передает найденное количество четных положительных чисел процессу 0. Процесс 0 принимает данные от всех процессов, суммирует их и выводит общее количество четных положительных чисел.

```
MPJ Express (0.44) is started in the multicore configuration
Process 0 (Thread ID: 23) full numbers: [-7, -91, 96, -77, -3, 20, -86, 26, 80, 73]
Process 0 (Thread ID: 23) local numbers: [-7, -91]
Process 0 found 0 even positive numbers.
Process 1 (Thread ID: 25) local numbers: [96, -77]
Process 1 found 1 even positive numbers.
Process 2 (Thread ID: 24) local numbers: [-3, 20]
Process 2 found 1 even positive numbers.
Process 3 (Thread ID: 22) local numbers: [-86, 26]
Process 3 found 1 even positive numbers.
Total even positive numbers: 3
```

Рис.3 – Результат выполнения задания №2

Листинг программ

Lab2Task1.java

```
import mpi.MPI;

public class Lab2Task1 {
    public static void main(String[] args) {
        MPI.Init(args);
        int rank = MPI.COMM_WORLD.Rank();
        int a = rank + 1;
        int b = (rank + 1) * 2;
        System.out.printf("Process %d (Thread ID: %d)\n", rank,
Thread.currentThread().getId());
        System.out.printf("    Initial values: a%d = %d, b%d = %d\n", rank, a, rank, b);

        int[] received_a = new int[1];
        int[] received_b = new int[1];
        int[] send_a_to = {2, 0, 3, 1};
        int[] send_b_to = {1, 3, 0, 2};
        int[] recv_a_from = {1, 3, 0, 2};
        int[] recv_b_from = {2, 0, 3, 1};
        if (rank == 0 || rank == 3) {
            sendData(a, b, send_a_to[rank], send_b_to[rank]);
            receiveData(received_a, received_b, recv_a_from[rank], recv_b_from[rank]);
        } else {
            receiveData(received_a, received_b, recv_a_from[rank], recv_b_from[rank]);
            sendData(a, b, send_a_to[rank], send_b_to[rank]);
        }

        System.out.println(" \n");
        System.out.printf("Process %d received: a = %d, b = %d\n", rank, received_a[0],
received_b[0]);
        System.out.printf("    Computed value c%d = a + b = %d + %d = %d", rank,
received_a[0], received_b[0], (received_a[0] + received_b[0]));

        MPI.Finalize();
    }

    private static void sendData(int a, int b, int send_a_to, int send_b_to) {
        MPI.COMM_WORLD.Send(new int[]{a}, 0, 1, MPI.INT, send_a_to, 99);
        MPI.COMM_WORLD.Send(new int[]{b}, 0, 1, MPI.INT, send_b_to, 99);
    }

    private static void receiveData(int[] received_a, int[] received_b, int recv_a_from, int
recv_b_from) {
        MPI.COMM_WORLD.Recv(received_a, 0, 1, MPI.INT, recv_a_from, 99);
        MPI.COMM_WORLD.Recv(received_b, 0, 1, MPI.INT, recv_b_from, 99);
    }
}
```

Lab2Task2.java

```
import mpi.MPI;
import java.util.Random;

public class Lab2Task2 {

    public static void main(String[] args) {
        MPI.Init(args);

        int size = MPI.COMM_WORLD.Size();
        int rank = MPI.COMM_WORLD.Rank();

        int numElements = 10;
        int elementsPerProcess = numElements / size;
        int[] numbers = new int[numElements];
        int[] localNumbers = new int[elementsPerProcess];

        if (rank == 0) {
            Random random = new Random();
            for (int i = 0; i < numElements; i++) {
                numbers[i] = random.nextInt(201) - 100;
            }

            System.out.printf("Process %d (Thread ID: %d) full numbers: %s\n", rank,
                Thread.currentThread().getId(), arrayToString(numbers));

            for (int i = 1; i < size; i++) {
                MPI.COMM_WORLD.Send(numbers, i * elementsPerProcess, elementsPerProcess,
MPI.INT, i, 99);
            }

            System.arraycopy(numbers, 0, localNumbers, 0, elementsPerProcess);
        } else {
            MPI.COMM_WORLD.Recv(localNumbers, 0, elementsPerProcess, MPI.INT, 0, 99);
        }

        System.out.printf("Process %d (Thread ID: %d) local numbers: %s\n", rank,
            Thread.currentThread().getId(), arrayToString(localNumbers));

        int localEvenCount = countPositive(localNumbers);
        System.out.printf("Process %d found %d even positive numbers.\n", rank,
localEvenCount);

        if (rank == 0) {
            int totalEvenCount = localEvenCount;
            int[] receivedCount = new int[1];

            for (int i = 1; i < size; i++) {
                MPI.COMM_WORLD.Recv(receivedCount, 0, 1, MPI.INT, i, 100);
```

```

        totalEvenCount += receivedCount[0];
    }

    System.out.printf("Total even positive numbers: %d\n", totalEvenCount);
} else {
    MPI.COMM_WORLD.Send(new int[]{localEvenCount}, 0, 1, MPI.INT, 0, 100);
}

MPI.Finalize();
}

private static int countPositive(int[] numbers) {
    int count = 0;
    for (int num : numbers) {
        if (num > 0 && num % 2 == 0) {
            count++;
        }
    }
    return count;
}

private static String arrayToString(int[] array) {
    StringBuilder sb = new StringBuilder("[");
    for (int i = 0; i < array.length; i++) {
        sb.append(array[i]);
        if (i < array.length - 1) {
            sb.append(", ");
        }
    }
    sb.append("]");
    return sb.toString();
}
}

```