

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

**ОТЧЕТ**  
**по лабораторной работе №01**  
**по дисциплине «Параллельные алгоритмы».**  
**Тема: Обмен сообщениями чётных и нечётных процессов.**

Студентка гр. 0382

Кривенцова Л.С.

Преподаватель

Татаринов Ю.С.

Санкт-Петербург

2022

### Задание.

Написать программу обмена сообщениями чётных и нечётных процессов. Замерить время на одну итерацию обмена и определите зависимость времени обмена от длины сообщения.

Листинг программы см. в Приложении А.

### Выполнение работы.

В программе осуществляется обмен сообщениями между процессами со следующей логикой: чётному процессу сказано отправлять сообщение с заданной длиной следующему по счёту, то есть нечётному процессу; аналогично даны указания нечётному процессу, с дополнительной проверкой, чтобы не выйти за рамки нумерации процессов.

Сообщения формируются заранее, под них освобождается количество памяти, соизмеримое с заданной длиной ( $int * message = (int *) (calloc(MAX\_LENGTH, sizeof(int)))$ ), где `MAX_LENGTH` – константа, которой предстоит изменение каждый запуск программы, чтобы выявить зависимости времени от длины передаваемого сообщения.

### Зависимость времени от длины сообщения.

Результат работы исходной программы рассматривался на 1..4 процессах.

Каждый запуск программы увеличим длину сообщения в 100 раз.

```
C:\Users\Serg>mpiexec -n 4 C:\Users\Serg\source\repos\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe
process 0, message length = 1, time = 0.000061
process 2, message length = 1, time = 0.000057
process 1, message length = 1, time = 0.000129
process 3, message length = 1, time = 0.000137
C:\Users\Serg>
```

Рис.1 - запуск программы с длиной сообщения = 1.

```
C:\Users\Serg>mpiexec -n 4 C:\Users\Serg\source\repos\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe
process 0, message length = 100, time = 0.000064
process 1, message length = 100, time = 0.000121
process 3, message length = 100, time = 0.000134
process 2, message length = 100, time = 0.000057
C:\Users\Serg>
```

Рис.2 - запуск программы с длиной сообщения = 100.

```
C:\Users\Serg>mpiexec -n 4 C:\Users\Serg\source\repos\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe
process 2, message length = 10000, time = 0.000199
process 1, message length = 10000, time = 0.000204
process 0, message length = 10000, time = 0.000201
process 3, message length = 10000, time = 0.000189
C:\Users\Serg>
```

Рис.3 - запуск программы с длиной сообщения = 10000.

```
C:\Users\Serg>mpiexec -n 4 C:\Users\Serg\source\repos\ConsoleApplication1\x64\Debug\ConsoleApplication1.exe
process 2, message length = 1000000, time = 0.000698
process 0, message length = 1000000, time = 0.000718
process 3, message length = 1000000, time = 0.000710
process 1, message length = 1000000, time = 0.000709
```

Рис.4 - запуск программы с длиной сообщения = 1 000 000.

Для выявления зависимости получим средние значения времени каждого запуска программы и получим данные:

length	1	100	10 000	1 000 000
time, [с]	(0.000061 + 0.000057 + 0.000129 + 0.000137)/4 = 0.000096	(0.000064 + 0.000121 + 0.000154 + 0.000057) / 100 = 0,000099	(0.000199 + 0.000204 + 0.000201 + 0.000189 )/4 = 0.00019825	( 0.000698 + 0.000718 + 0.000710 + 0.000709 )/4 = 0.0007065



## ПРИЛОЖЕНИЕ А

### ИСХОДНЫЙ КОД ПРОГРАММЫ

```
#include <stdio.h>
#include <mpi.h>
#include <cstdlib>
#define MAX_LENGTH 1

int main(int argc, char* argv[])
{
    int size, rank;
    double time_start, time;
    MPI_Status status;
    int* message_1 = (int*)(calloc(MAX_LENGTH, sizeof(int)));
    int* message_2 = (int*)(calloc(MAX_LENGTH + 10, sizeof(int)));
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    time_start = MPI_Wtime();
    if ((rank % 2) == 0 && rank >= 0) {
        if (rank < size - 1)
            MPI_Send(message_1, MAX_LENGTH, MPI_INT, rank + 1, 0,
                     MPI_COMM_WORLD);
    }
    else if (rank >= 0)
        MPI_Recv(message_2, MAX_LENGTH, MPI_INT, rank - 1, 0,
MPI_COMM_WORLD,
                &status);

    time = MPI_Wtime() - time_start;
    printf("\nprocess %d, message length = %d, time = %f", rank,
MAX_LENGTH, time);
    MPI_Finalize();
    return 0;
}
```