

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 четных и нечетных процессов

Студентка гр. 9382

Преподаватель

Иерусалимов Н.

Татаринов Ю.С.

Санкт-Петербург

2021

Цель работы

Научиться управлять взаимодействием разных процессов, замерять время работы процессов.

Формулировка задания

Задание: Напишите программу обмена сообщениями чётных и нечётных процессов. Замерьте время на одну итерацию обмена и определите зависимость времени обмена от длины сообщения.

```
#include <stdio.h>
#include "mpi.h"
int main(int argc, char* argv[]) {
    int ProcNum, ProcRank, RecvRank;
    double start, finish;
    MPI_Status Status;
    MPI_Init(&argc, &argv);
    MPI_Comm_size(MPI_COMM_WORLD, &ProcNum);
    MPI_Comm_rank(MPI_COMM_WORLD, &ProcRank);
    if (ProcRank % 2 == 0) {
        start= MPI_Wtime();
        if (ProcRank < ProcNum - 1) {
            printf("\n To ->%3d", ProcRank);
            MPI_Recv(&RecvRank, 1, MPI_INT, ProcRank + 1, MPI_ANY_TAG,
MPI_COMM_WORLD, &Status);
            printf("\n From <-%3d", RecvRank);
        }
        if (ProcRank == ProcNum - 1) {
            printf("\n Single process %3d", ProcRank);
        }
    }
```

```

    finish = MPI_Wtime();

    printf("\n Process %d: time request=%lf\n", ProcRank, (finish - start) / 100);
}
else
    MPI_Send(&ProcRank, 1, MPI_INT, ProcRank - 1, 0,
MPI_COMM_WORLD);
    MPI_Finalize();
    return 0;
}

```



```

nereus@Nereus:~/CLionProjects/untitled1$ mpirun -np 8 ./a.out
Invalid MIT-MAGIC-COOKIE-1 key
To -> 4
From <- 5
Process 4: time request=0.000000

To -> 0
From <- 1
Process 0: time request=0.000001

To -> 2
From <- 3
Process 2: time request=0.000001

To -> 6
From <- 7
Process 6: time request=0.000001
nereus@Nereus:~/CLionProjects/untitled1$

```

рис. 1 — результат работы программы на N процессорах

Нечетные процессы отправляют сообщения, а четные – их принимают. Если же процессору не хватает пары то выводится его номер с приставкой Single

Разные длины пересылаемых сообщений

Выполнение программы не зависит от числа процессоров, так как работа происходит попарно. При увеличении длины сообщения процесс обмена происходит медленнее на незначительное количество времени.

```
#include <stdio.h>
```

```
#include "mpi.h"
```

```
int main(int argc, char* argv[]) {
```

```
    int ProcNum, ProcRank, RecvRank;
```

```
    double time_start, time_finish;
```

```
    int N = 50 000;
```

```
    char text[N + 1];
```

```
    for (int i = 0; i < N; i++)
```

```
        if(i%2==0) text[i] = '-';
```

```
        else text[i] = '_';
```

```
    text[N] = '\0';
```

```
    MPI_Status Status;
```

```
    MPI_Init(&argc, &argv);
```

```
    MPI_Comm_size(MPI_COMM_WORLD, &ProcNum);
```

```
    MPI_Comm_rank(MPI_COMM_WORLD, &ProcRank);
```

```
    if (ProcRank % 2 == 0) {
```

```
        time_start = MPI_Wtime();
```

```
        if (ProcRank < ProcNum - 1) {
```

```
            printf("\n To -> %3d", ProcRank);
```

```
            MPI_Recv(&RecvRank, 1, MPI_INT, ProcRank + 1,
```

```
                    MPI_ANY_TAG, MPI_COMM_WORLD, &Status);
```

```
            printf("\n From <- %3d", RecvRank);
```

```
            MPI_Recv(text, N + 1, MPI_CHAR, ProcRank + 1, MPI_ANY_TAG,
```

```

MPI_COMM_WORLD, &Status);

    printf("\n Get request %s", text);
} if (ProcRank == ProcNum - 1) {
    printf("\n Single %3d", ProcRank);
}

time_finish = MPI_Wtime();


printf("\n Process %d: time request=%lf\n", ProcRank, (time_finish -
time_start) / 100);
}
else {
    MPI_Send(&ProcRank, 1, MPI_INT, ProcRank - 1, 0,
MPI_COMM_WORLD);

    MPI_Send(text, N + 1, MPI_CHAR, ProcRank - 1, 0,
MPI_COMM_WORLD);
}

MPI_Finalize();

return 0;

```



```

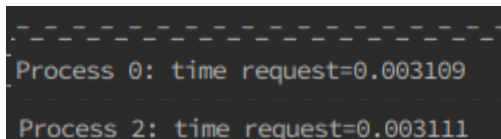
nereus@Nereus:~/CLionProjects/untitled1$ mpirun -np 4 ./a.out
Invalid MIT-MAGIC-COOKIE-1 key

To -> 2
From <- 3
Get request _ _ _ _ _
Process 2: time request=0.000000
To -> 0
From <- 1
Get request _ _ _ _ _
Process 0: time request=0.000001
nereus@Nereus:~/CLionProjects/untitled1$

```

Рис. 2 — скорость при 10 символах

Разницу в скорости смог получить только на 50 000 символах



```
-----  
Process 0: time request=0.003109  
Process 2: time request=0.003111
```

Рис. 3 — скорость при 50 000 символах

Выводы

В ходе лабораторной работы была написана и модифицирована программа MPI для обмена сообщениями между четными и нечетными процессами. Чем больше длина сообщения, тем больше затрачивается времени на обмен.

