МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Системы параллельной обработки данных»

Тема: ИСПОЛЬЗОВАНИЕ АРГУМЕНТОВ-ДЖОКЕРОВ

Студент гр. 5304 _____ Лянгузов А.А.

Преподаватель _____ Татаринов Ю.С.

Санкт-Петербург

2019

Задание 1. Пляжный волейбол:

Процесс 0 генерирует сообщение и посылает его любому другому процессу группы. Процесс-приемник передает сообщение дальше. Выбор процесса-адресата осуществляется случайным образом.

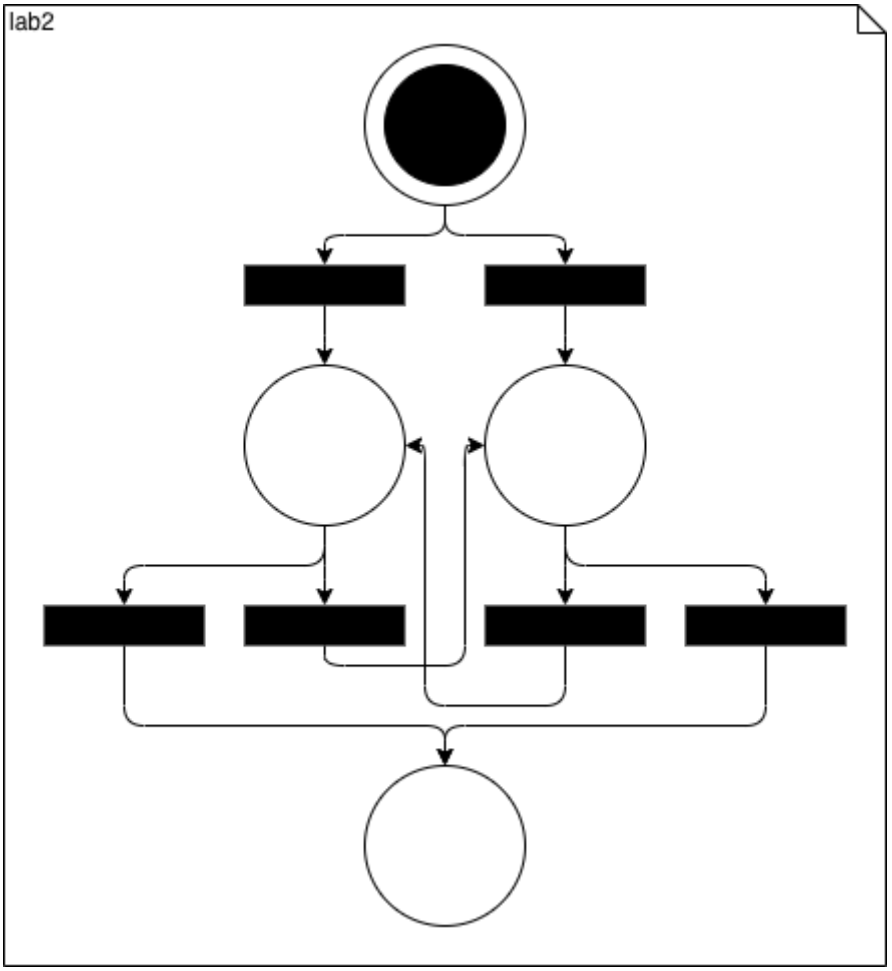
Выполнение работы

В отличие от первой работы, при передаче сообщения в данной работе, не учитывается, было ли послано сообщение процессу ранее. Процесс для приема сообщения выбирается случайно среди всех процессов (в том числе и главного). Как только “мяч” оказывается у главного процесса, программа завершает свою работу.

В функции *MPI_Recv(&RecvRank, 1, MPI_INT, MPI_ANY_SOURCE, MPI_ANY_TAG, MPI_COMM_WORLD, &Status);*

используется джокер MPI_ANY_SOURCE. То есть будем принимать сообщения от любого процесса. Это нужно для случайной передачи “мяча” любому из процессов группы.

Сеть Петри



Изображена сеть петри для трех процессов. Была выбрана для большей понятности архитектуру сети. Данную архитектуру легко можно масштабировать для любого числа процессов.

Пример выполнения программы

```
(base) xtail@~/Projects/LETI/Parallel/ParallelLabs$ ./assembly.sh launch lab2 3
  1 catch ball from   2
  1 throw ball to   2

  1 catch ball from   2
  1 throw ball to   2

  1 catch ball from   2
  1 throw ball to   2

  2 catch ball from   0
  2 throw ball to   1

  2 catch ball from   1
  2 throw ball to   1

  2 catch ball from   1
  2 throw ball to   1

  2 catch ball from   1
  2 throw ball to   0

  0 throw ball to   2

(base) xtail@~/Projects/LETI/Parallel/ParallelLabs$
```

Рис.1.Результат работы программы.

Выводы

В ходе выполнения лабораторной работы ознакомились с применением джокеров путем написания программы имитации игры в пляжный волейбол (без сетки).

ПРИЛОЖЕНИЕ А. КОД ПРОГРАММЫ

```
#include <stdlib.h>
#include <stdio.h>
#include "mpi.h"
int main(int argc, char* argv[])
{
   int ProcNum, ProcRank, RecvRank;
   MPI_Status Status;
   MPI_Init(&argc, &argv); //parallel part of app start
   MPI_Comm_size(MPI_COMM_WORLD, &ProcNum); //declare size of processes
(group id, group size(return))
   MPI_Comm_rank(MPI_COMM_WORLD, &ProcRank); //define process rank in
groud (group id, rank(return))
   if(ProcRank == 0)
   {
       int NextProcRank = rand() % (ProcNum - 1) + 1;
       printf("%3d throw ball to %3d \n\n", ProcRank, NextProcRank);
       MPI_Send(&ProcRank, 1, MPI_INT, NextProcRank, 0, MPI_COMM_WORLD);
       MPI_Recv(&RecvRank, 1, MPI_INT, MPI_ANY_SOURCE, MPI_ANY_TAG,
MPI_COMM_WORLD, &Status);
       for(int i = 1; i < ProcNum; i++)
       {
           int stop = -1;
           MPI_Send(&stop, 1, MPI_INT, i, 0, MPI_COMM_WORLD);
       }
   }
   else
   {
       for(;;)
       {
           MPI_Recv(&RecvRank, 1, MPI_INT, MPI_ANY_SOURCE, MPI_ANY_TAG,
MPI_COMM_WORLD, &Status);
           if(RecvRank == -1)
           {
               break;
           }

           printf("%3d catch ball from %3d \n", ProcRank, RecvRank);
```

```
            int NextProcRank = rand() % (ProcNum - 1);
            if(NextProcRank == ProcRank)
            {
                NextProcRank = NextProcRank + 1;
            }
            printf("%3d throw ball to %3d \n\n", ProcRank, NextProcRank);
            MPI_Send(&ProcRank, 1, MPI_INT, NextProcRank, 0,
MPI_COMM_WORLD);
        }
    }
    MPI_Finalize(); //parallel part of app finish
    return 0;
}
```