**PARALLEL AND DISTRIBUTED COMPUTING LAB**

**REPORT**

**NAME:** S Shyam Sundaram

**REG NO:** 19BCE1560

**PROGRAMMING ENVIRONMENT:** MPI

**PROBLEM:** MPI

**DATE:** 10th November, 2021

**HARDWARE CONFIGURATION:**

|  |  |  |  |
| --- | --- | --- | --- |
| CPU NAME | | : | Intel core i5 – 1035G1 @ 1.00 Ghz |
| Number of Sockets: | | : | 1 |
| Cores per Socket | | : | 4 |
| Threads per core | | : | 1 |
| L1 | Cache size | : | 320KB |
| L2 | Cache size | : | 2MB |
| L3 | Cache size (Shared): | | 6MB |
| RAM | | : | 8 GB |

**QUESTION**

Write an MPI program that finds the maximum and minimum element in a matrix using MPI\_Reduce, MPI\_MIN and MPI\_MAX.

**CODE**

#include <mpi.h>

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

const int n = 8; //must be equal to number of processes -np while running with mpirun

const int m = 5;

void createMatrix(int matrix[n][m])

{

    int c=0;

    for (int i = 0; i < n; i++)

    {

        for (int j = 0; j < m; j++)

        {

            printf("%d\t",c);

            matrix[i][j] = c++;

        }

        printf("\n");

    }

}

int main(int argc, char\* argv[])

{

    int id = 0;

    int comm\_size = 0;

    int mat[n][m]; //the matrix created and distributed by master process

    int row[m]; //one row of matrix to be used by each process

    int colmax[m]; //will contain the maximum of each column

    int colmin[m]; //will contain the minimum of each column

    MPI\_Init(&argc, &argv);

    MPI\_Comm\_rank(MPI\_COMM\_WORLD, &id);

    MPI\_Comm\_size(MPI\_COMM\_WORLD, &comm\_size);

    if (id == 0)

    {

        createMatrix(mat);

        printf("Name: Shyam S\nReg num: 19BCE1560\n\n");

    }

    MPI\_Scatter(mat, m, MPI\_INT, row, m, MPI\_INT, 0, MPI\_COMM\_WORLD);

    MPI\_Reduce(row, colmax, m, MPI\_INT, MPI\_MAX, 0, MPI\_COMM\_WORLD); //colmax has maximum element of each column

    MPI\_Reduce(row, colmin, m, MPI\_INT, MPI\_MIN, 0, MPI\_COMM\_WORLD); //colmin has minimum element of each column

    // if(id!=0)

    // {

    //     for(int i=0;i<m;++i)

    //     {

    //         printf("process %d element %d = %d\n",id,i,row[i]);

    //     }

    // }

    if (id == 0)

    {

        int max = colmax[0];

        for(int i=1;i<m;++i)

        {

            if(max<colmax[i])

            max=colmax[i];

        }

        int min = colmin[0];

        for(int i=1;i<m;++i)

        {

            if(min>colmin[i])

            min=colmin[i];

        }

        printf("Max: %d\nMin: %d\n",max,min);

    }

    MPI\_Finalize();

    return 0;

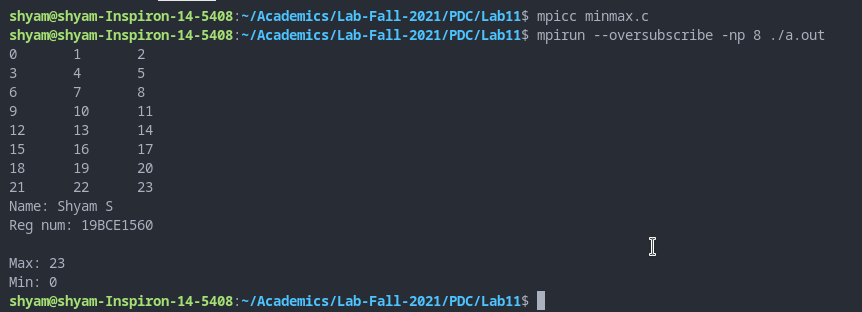
}

**COMMANDS**

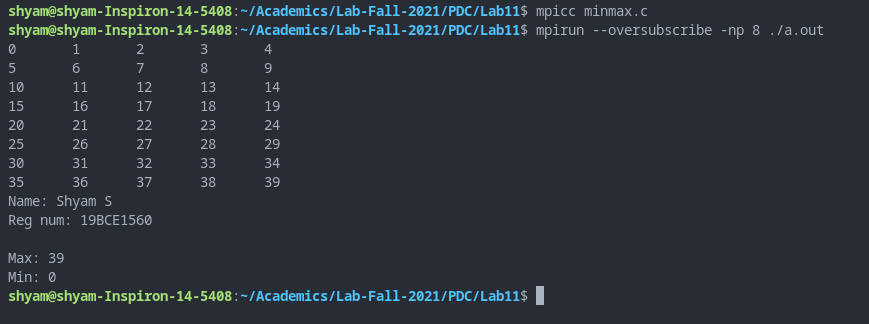
mpicc matrix.c

mpirun --oversubscribe -np 8 ./a.out

**OUTPUT**

****

With n=8 and m=3



With n=8 and m=5

**OBSERVATION**

Each worker gets a row of the matrix. Then, MPI\_Reduce finds the max and min in each column by comparing the elements in same index of each row belonging to different worker. We get the maximum and minimum of each column this way. Then using a function, we just loop through the maximum and minimum of each column and find the maximum and minimum among them. (Time shown below were taken with some additional code)

|  |  |  |
| --- | --- | --- |
| **NxM** | **NUMBER OF PROCESSES** | **TIME** |
| 100x100 | 1 | 3.60 |
| 2 | 0.25 |
| 4 | 0.00 |
| 100x1000 | 1 | 10.3 |
| 2 | 6.32 |
| 4 | 1.34 |

**CONCLUSION**

We have found maximum and minimum elements of a matrix using MPI\_MIN and MPI\_MAX in C.