Parallel Programming

REPORT PW8

Nafila Amirli

Exercise 1

Write a parallel MPI program that computes the maximum of elements in an array. The process 0 initializes the array with random numbers.

The array is distributed using MPI_Scatter.

```
//array is distributed
MPI_Scatter(initData, block_length, MPI_INT, data, block_length, MPI_INT, 0, MPI_COMM_WORLD);
```

Each process computes a local max.

```
//computing the local max of each process
for (i = 0; i < block_length; i++)
    localMax = data[(largest(data, size))];
printf("procces %d local Max= %d\n", rank, localMax);
```

Results are collected by process 0 using MPI_Gather.

```
//Results are collected by process 0 using MPI_Gather
MPI_Gather(&localMax, 1, MPI_INT, arr, 1, MPI_INT, 0, MPI_COMM_WORLD);
```

Process 0 computes and displays the global max.

```
if (rank == 0)
{
    max = largest(arr, size);
    printf("\nGlobal Max = %d\n", max);
```

Terminal output:

```
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW8$ mpicc ex1.c
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW8$ mpiexec -n 4 ./a.out
procces 0 local Max= 3
procces 1 local Max= 0
procces 2 local Max= 75
procces 3 local Max= 0

Global Max = 75
```

Exercise 2

Re-Write the parallel program that computes the max of elements in an array by using MPI_Reduce.

Different part of code:

```
}
//maximum for each process
max = largest(arr, len);
printf("p%d  max = %d\n", rank, max);
//Global maximum
MPI_Reduce(&max, &GlobMax, 1, MPI_INT, MPI_MAX, 0, MPI_COMM_WORLD);
if (rank == 0)
{
    printf("\nMPI_Reduce the Global Max %d\n", GlobMax);
```

Terminal output:

```
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW8$ mpicc ex2.c
nafila@nafila-Lenovo-V110-15ISK:~/paralel/PW8$ mpiexec -n 4 ./a.out
p0  max = 99

MPI_Reduce the Global Max 399
p1  max = 199
p2  max = 299
p3  max = 399
```