

# CSE 4001

## PARALLEL AND DISTRIBUTED COMPUTING



### Lab Assessment – 4

L27+L28 | PLBG04

Dr. Narayanan Prasanth

FALL SEMESTER 2021-22

by

**SHARADINDU ADHIKARI**

19BCE2105

**Q1. Develop an MPI program to compute the average of numbers using the following functions and compute the time required for the same.**

**(a) MPI\_Scatter and MPI\_Gather**

**Code:**

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

int main(int argc, char** argv)
{

    printf("\nSharadindu Adhikari, 19BCE2105 \n");
    printf("\n \n"); //for some space

    int my_rank;
    int total_processes;
    int root = 0;
    int data[100];
    int data_loc[100];
    float final_res[100];

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &my_rank);
    MPI_Comm_size(MPI_COMM_WORLD, &total_processes); //total number of processes

    int input_size = 0;
    if (my_rank == 0){
        printf("Input how many numbers: ");
        scanf("%d", &input_size);

        printf("Input the elements of the array: ");
        for(int i=0; i<input_size; i++){
            scanf("%d", &data[i]);
        }
    }

    MPI_Bcast(&input_size, 1, MPI_INT, root, MPI_COMM_WORLD); //broadcast the
input size to all processes

    int loc_num = input_size/total_processes;

    MPI_Scatter(&data, loc_num, MPI_INT, data_loc, loc_num, MPI_INT, root,
MPI_COMM_WORLD); //scatter the data to all processes
```

```
int loc_sum = 0;
for(int i=0; i< loc_num; i++)
    loc_sum += data_loc[i];
float loc_avg = (float) loc_sum / (float) loc_num;
MPI_Gather(&loc_avg, 1, MPI_FLOAT, final_res, 1, MPI_FLOAT, root,
MPI_COMM_WORLD); //gather the data from all processes

if(my_rank==0){
    float fin = 0;
    for(int i=0; i<total_processes; i++)
        fin += final_res[i];
    float avg = fin / (float) total_processes; //calculate the average
    printf("Final average: %f \n", avg);
}
MPI_Finalize();
return 0;
}
```

## Input & Output:

The screenshot displays the Visual Studio Code interface with a file explorer on the left showing a project structure with folders 'Lab1', 'Lab2', 'Lab4', and files '1a.c' and 'Midterm'. The main editor shows the source code for '1a.c', which includes headers for `<stdio.h>` and `"mpi.h"`, and defines a `main` function. The code initializes MPI, gathers data from all processes, and calculates the final average. The terminal at the bottom shows the execution of the program, including the compilation command `mpicc 1a.c -o 1a` and the output of the program, which prompts for the number of elements and their values, and then prints the final average.

```
1 #include <stdio.h>
2 #include "mpi.h"
3
4 int main(int argc, char** argv)
5 {
6
7     printf("\nSharadindu Adhikari, 19BCE2105 \n");
8     printf("\n \n"); //for some space
9
10    int my_rank;
11    int total_processes;
12    int root = 0;
13    int data[100];
14    int data_loc[100];
15    float final_res[100];
16
17    MPI_Init(&argc, &argv);
18    MPI_Comm_rank(MPI_COMM_WORLD, &my_rank);
19    MPI_Comm_size(MPI_COMM_WORLD, &total_processes); //total number of processes
20
21    shara-d@Rohans-Workstation:~/PDC$ cd Lab4
22    shara-d@Rohans-Workstation:~/PDC/Lab4$ mpicc 1a.c -o 1a
23    shara-d@Rohans-Workstation:~/PDC/Lab4$ ./1a
24
25    Sharadindu Adhikari, 19BCE2105
26
27    Input how many numbers: 17
28    Input the elements of the array: 2 4 0 1 98 7 63 8 44 5 8 10 67 99 23 37 3
29    Final average: 28.176470
30    shara-d@Rohans-Workstation:~/PDC/Lab4$
```

## (b) MPI\_Scatter and MPI\_Reduce

### Code & Input:

```
#include <stdio.h>
#include <stdlib.h>
#include <mpi.h>
#include <assert.h>

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

    printf("\nSharadindu Adhikari, 19BCE2105 \n");
    printf("\n \n"); //for some space

    int world_rank, world_size, int_size, i ;
    int avg, rand_nums,n , sub_avg, startval, endval, *sub_rand_nums, *sub_avgs,
N=9 ;
    MPI_Init( &argc , &argv );
    MPI_Comm_size(MPI_COMM_WORLD , &world_size); //get the number of processes
    MPI_Comm_rank(MPI_COMM_WORLD , &world_rank); //get the rank of the process
    startval = N * world_rank/world_size + 1;
    endval = N * (world_rank+1) / world_size;

    n = N / world_size;

    // Sum the numbers locally
    int local_sum = 0;

    for (i = startval; i <= endval; i++) {
        local_sum = local_sum+i;
    }

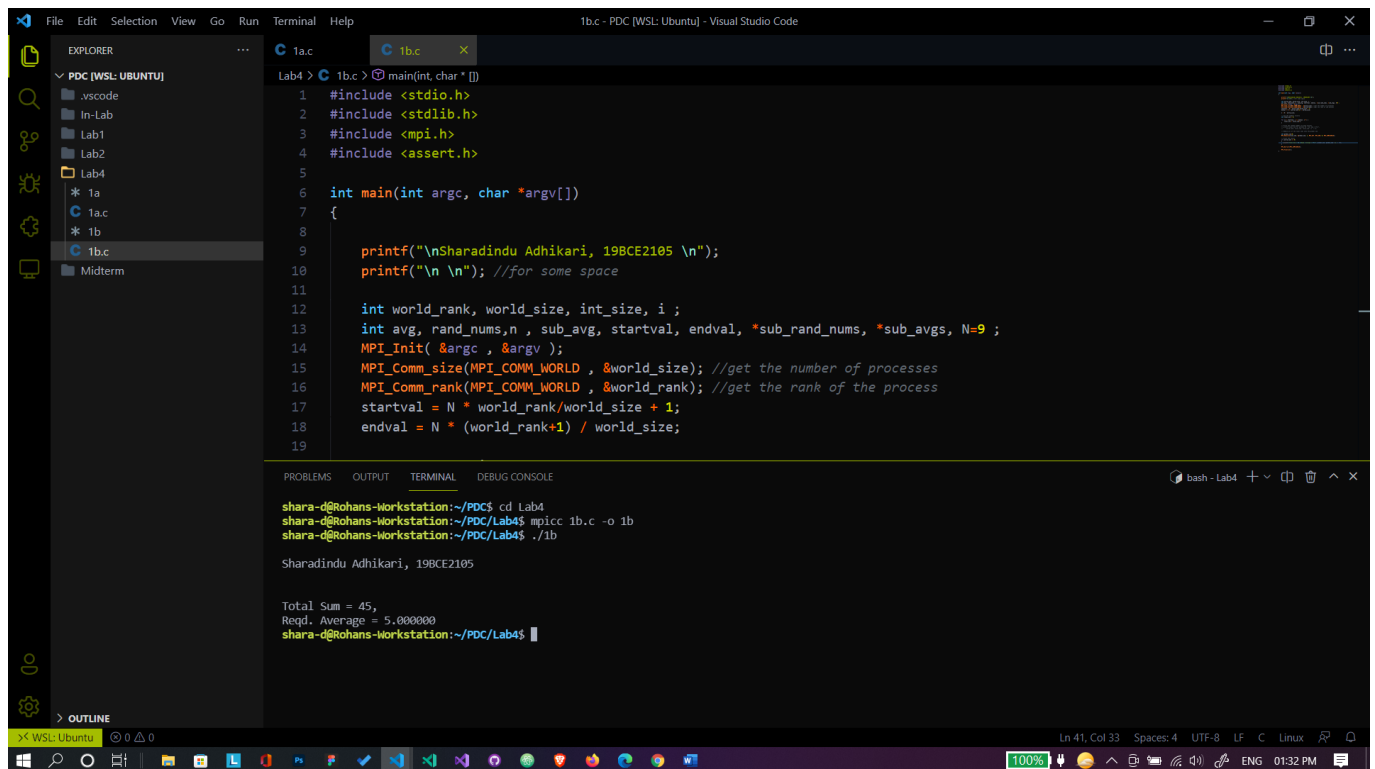
    int global_sum=0;
    MPI_Reduce(&local_sum, &global_sum, 1, MPI_INT, MPI_SUM, 0, MPI_COMM_WORLD);

    // Print the result
    if (world_rank == 0)
    {
        printf("Total Sum = %d,\nReqd. Average = %f\n", global_sum, global_sum *
1. / N);
    }
}
```

```
MPI_Barrier(MPI_COMM_WORLD);

MPI_Finalize();
}
```

## Output:



```
Lab4 > C 1b.c > main(int, char* [])
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <mpi.h>
4 #include <assert.h>
5
6 int main(int argc, char *argv[])
7 {
8
9     printf("\nSharadindu Adhikari, 19BCE2105 \n");
10    printf("\n \n"); //for some space
11
12    int world_rank, world_size, int_size, i ;
13    int avg, rand_nums, n , sub_avg, startval, endval, *sub_rand_nums, *sub_avgs, N=9 ;
14    MPI_Init( &argc , &argv );
15    MPI_Comm_size(MPI_COMM_WORLD , &world_size); //get the number of processes
16    MPI_Comm_rank(MPI_COMM_WORLD , &world_rank); //get the rank of the process
17    startval = N * world_rank/world_size + 1;
18    endval = N * (world_rank+1) / world_size;
19
20    PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE
21
22    shara-d@Rohans-Workstation:~/PDC$ cd Lab4
23    shara-d@Rohans-Workstation:~/PDC/Lab4$ mpicc 1b.c -o 1b
24    shara-d@Rohans-Workstation:~/PDC/Lab4$ ./1b
25
26    Sharadindu Adhikari, 19BCE2105
27
28    Total Sum = 45,
29    Req'd. Average = 5.000000
30    shara-d@Rohans-Workstation:~/PDC/Lab4$
```

**Alternative Solution to Q1:**

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <mpi.h>

float *gen_random(int num_elements)
{
    float *nums = (float *)malloc(sizeof(float) * num_elements);
    int i;
    for (i = 0; i < num_elements; i++)
    {
        nums[i] = (rand() % 10000);
    }

    return nums;
}

float calculate_avg(float *array, int num_elements)
{
    float avg = 0;
    int i;
    for (i = 0; i < num_elements; i++)
    {
        avg += array[i];
    }
    avg = avg / num_elements;
    return avg;
}

int main(int argc, char **argv)
{
    srand(time(NULL));
    int ne_proc = 10000;
    double start, end;

    MPI_Init(NULL, NULL);
    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);
    MPI_Barrier(MPI_COMM_WORLD);
```

```
start = MPI_Wtime();

float *rand_nums = NULL;
if (world_rank == 0)
{
    printf("\nSharadindu Adhikari\n");
    printf("19BCE2105\n\n");

    rand_nums = gen_random(ne_proc * world_size);
}

float *sub_rand_nums = (float *)malloc(sizeof(float) * ne_proc);

MPI_Scatter(rand_nums, ne_proc, MPI_FLOAT, sub_rand_nums, ne_proc,
MPI_FLOAT, 0, MPI_COMM_WORLD);

float sub_avg = calculate_avg(sub_rand_nums, ne_proc);

float sub_avgs = 0;

MPI_Reduce(&sub_avg, &sub_avgs, 1, MPI_FLOAT, MPI_SUM, 0,
MPI_COMM_WORLD);

if (world_rank == 0)
{
    float avg = sub_avgs/world_size;
    printf("Avg of all elements = %f\n\n", avg);
}

if (world_rank == 0)
{
    free(rand_nums);
}
free(sub_rand_nums);

MPI_Barrier(MPI_COMM_WORLD);
end = MPI_Wtime();

if (world_rank == 0)
{
    printf("Time - %lf seconds\n\n", end-start);
}
MPI_Finalize();
}
```

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #include <mpi.h>
5
6  float *gen_random(int num_elements)
7  {
8      float *nums = (float *)malloc(sizeof(float) * num_elements);
9      int i;
10     for (i = 0; i < num_elements; i++)
11     {
12         nums[i] = (rand() % 10000);
13     }
14
15     return nums;
16 }
17
18 float calculate_avg(float *array, int num_elements)
19 {
20     float avg = 0;
21     int i;
22     for (i = 0; i < num_elements; i++)
23     {
24         avg += array[i];
25     }
26     avg = avg / num_elements;
27     return avg;
28 }
29
30 int main(int argc, char **argv)
31 {
32     srand(time(NULL));
33     int ne_proc = 10000;

```

## OUTPUT:

```

shara-d@Rohans-Workstation:~/PDC$ cd Lab4
shara-d@Rohans-Workstation:~/PDC/Lab4$ mpicc prob1.c -o prob1
shara-d@Rohans-Workstation:~/PDC/Lab4$ mpirun -np 4 ./prob1

WARNING: Linux kernel CMA support was requested via the
btl_vader_single_copy_mechanism MCA variable, but CMA support is
not available due to restrictive ptrace settings.

The vader shared memory BTL will fall back on another single-copy
mechanism if one is available. This may result in lower performance.

Local host: Rohans-Workstation

-----
Sharadindu Adhikari
19BCE2105

Avg of all elements = 4990.779297

Time - 0.147402 seconds

[Rohans-Workstation:02913] 3 more processes have sent help message help-btl-vader.txt / cma-permission-denied
[Rohans-Workstation:02913] Set MCA parameter "orte_base_help_aggregate" to 0 to see all help / error messages
shara-d@Rohans-Workstation:~/PDC/Lab4$

```



**Q2.**

**2. Develop a MPI program to find the occurrence of a number in a given set (size not less than 20). Use collective communication to implement the same and print the time taken to complete the process. (4)**

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <mpi.h>

int *gen_random(int num_elements)
{
    int *nums = (int *)malloc(sizeof(int) * num_elements);
    int i;
    for (i = 0; i < num_elements; i++)
    {
        nums[i] = (rand() % 100);
    }
    return nums;
}

int main(int argc, char **argv)
{
    srand(time(NULL));
    int ne_proc = 1000;
    double start, end;

    MPI_Init(NULL, NULL);
    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);
    MPI_Barrier(MPI_COMM_WORLD);
    start = MPI_Wtime();

    int *rand_nums = NULL;
    int num;
    int count = 0;
    if (world_rank == 0)
    {
        printf("\nSharadindu Adhikari\n");
    }
}
```

```
    printf("19BCE2105\n\n");

    rand_nums = gen_random(ne_proc * world_size);
    num = rand() % 100;
}

int *sub_rand_nums = (int *)malloc(sizeof(int) * ne_proc);

MPI_Scatter(rand_nums, ne_proc, MPI_INT, sub_rand_nums, ne_proc,
MPI_INT, 0, MPI_COMM_WORLD);
MPI_Bcast(&num, 1, MPI_INT, 0, MPI_COMM_WORLD);
MPI_Barrier(MPI_COMM_WORLD);

for (int i = 0; i < ne_proc; i++)
{
    if (sub_rand_nums[i] == num)
    {
        count++;
        printf("Element %d found at index : %d\n", num, world_rank *
ne_proc + i);
    }
}

if (world_rank == 0)
{
    free(rand_nums);
}
free(sub_rand_nums);

MPI_Barrier(MPI_COMM_WORLD);
end = MPI_Wtime();

if (world_rank == 0)
{
    printf("\nTime - %lf seconds\n\n", end - start);
}
MPI_Finalize();
}
```

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #include <mpi.h>
5
6  int *gen_random(int num_elements)
7  {
8      int *nums = (int *)malloc(sizeof(int) * num_elements);
9      int i;
10     for (i = 0; i < num_elements; i++)
11     {
12         nums[i] = (rand() % 100);
13     }
14     return nums;
15 }
16
17 int main(int argc, char **argv)
18 {
19     srand(time(NULL));
20     int ne_proc = 1000;
21     double start, end;
22
23     MPI_Init(NULL, NULL);
24     int world_rank;
25     MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
26     int world_size;
27     MPI_Comm_size(MPI_COMM_WORLD, &world_size);
28     MPI_Barrier(MPI_COMM_WORLD);
29     start = MPI_Wtime();
30
31     int *rand_nums = NULL;
32     int num;
33     int count = 0;
34     if (world_rank == 0)

```

## FINAL OUTPUT:

```

shara-d@Rohans-Workstation:~/PDC$ cd Lab4
shara-d@Rohans-Workstation:~/PDC/Lab4$ mpicc prob2.c -o prob2
shara-d@Rohans-Workstation:~/PDC/Lab4$ mpirun -np 4 ./prob2

WARNING: Linux kernel CMA support was requested via the
btl_vader_single_copy_mechanism MCA variable, but CMA support is
not available due to restrictive ptrace settings.

The vader shared memory BTL will fall back on another single-copy
mechanism if one is available. This may result in lower performance.

Local host: Rohans-Workstation
[Rohans-Workstation:02246] 2 more processes have sent help message help-btl-vader.txt / cma-permission-denied
[Rohans-Workstation:02246] Set MCA parameter "orte_base_help_aggregate" to 0 to see all help / error messages

Sharadindu Adhikari
19BCE2105

Element 89 found at index : 3050
Element 89 found at index : 3291
Element 89 found at index : 3547
Element 89 found at index : 3598
Element 89 found at index : 3611
Element 89 found at index : 3670
Element 89 found at index : 3761
Element 89 found at index : 3804
Element 89 found at index : 3860
Element 89 found at index : 3984
Element 89 found at index : 24
Element 89 found at index : 54
Element 89 found at index : 79
Element 89 found at index : 235
Element 89 found at index : 386
Element 89 found at index : 446
Element 89 found at index : 480
Element 89 found at index : 842
Element 89 found at index : 863
Element 89 found at index : 895
Element 89 found at index : 945

Time - 0.088562 seconds

Element 89 found at index : 1273
Element 89 found at index : 1318
Element 89 found at index : 1355
Element 89 found at index : 1735
Element 89 found at index : 1746
Element 89 found at index : 1978
Element 89 found at index : 2166
Element 89 found at index : 2448
Element 89 found at index : 2628
Element 89 found at index : 2685
[Rohans-Workstation:02246] 1 more process has sent help message help-btl-vader.txt / cma-permission-denied
shara-d@Rohans-Workstation:~/PDC/Lab4$

```