**PARALLEL AND DISTRIBUTED COMPUTING LAB**

**REPORT**

**NAME:** S Shyam Sundaram

**REG NO:** 19BCE1560

**PROGRAMMING ENVIRONMENT:** MPI

**PROBLEM:** MPI

**DATE:** 13th October, 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 calculates the sum of numbers read from a text file. Note: Text file is named num.txt.

**CODE**

#include <mpi.h>

#include<stdio.h>

#define MAXSIZE 100 //the text file has just 100 numbers

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

{

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

    MPI\_Init(&argc,&argv);

    float startwtime,endwtime,totalTime;

    int myid;

    long int s,s0,startIndex,endIndex;

    long int i=0;

    float sum,part\_sum;

    float data[MAXSIZE];

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

    int numprocs;

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

    int namelen;

    char processor\_name[MPI\_MAX\_PROCESSOR\_NAME];

    MPI\_Get\_processor\_name(processor\_name, &namelen);

    FILE \*fp;

    fprintf(stderr,"Process %d is on %s\n",myid,processor\_name);

    fflush(stderr);

    FILE \*fptr;

    int num;

    fptr=fopen("num.txt","r");

    while((num=getw(fptr))!=EOF)

    {

        data[i++]=num/1.0;

    }

    fclose(fptr);

    if(myid==0) //master

    {

        s=(int)(MAXSIZE/numprocs);

        s0=s+(MAXSIZE%numprocs);

        printf("s=%ld,s0=%ld \n",s,s0);

    }

    else //worker

    {

        s=(int)(MAXSIZE/numprocs);

        s0=s+(MAXSIZE%numprocs);

        startIndex = s0+(myid-1)\*s;

        endIndex=startIndex+s;

        }

    //Broadcasting

    MPI\_Bcast(&s,1,MPI\_FLOAT,0,MPI\_COMM\_WORLD);

    MPI\_Bcast(&s0,1,MPI\_FLOAT,0,MPI\_COMM\_WORLD);

    //Time

    totalTime=0;

    if(myid==0)

    {

        startwtime=MPI\_Wtime();

    }

    //Partial Sum

    if(myid==0)

    {

        //master partial sum

        for(i=0;i<s0;i++)

        {

            part\_sum = part\_sum+data[i];

        }

        printf("Partial Sum from master = %f from processor %d\n", part\_sum,myid);

    }

    else

    {

        //worker partial sum

        for(i=startIndex;i<endIndex;i++)

        {

            part\_sum=part\_sum+data[i];

        }

        printf("Partial Sum from worker = %f from processor %d\n", part\_sum,myid);

    }

    //Global Sum

    MPI\_Reduce(&part\_sum,&sum,1, MPI\_FLOAT,MPI\_SUM,0,MPI\_COMM\_WORLD);

    if(myid==0)

    {

        endwtime=MPI\_Wtime();

        totalTime=endwtime-startwtime;

        printf("Global Sum is %f\n", sum);

        printf("Execution time %f sec\n",totalTime);

    }

    MPI\_Finalize();

}

**CODE (for write.c that created the num.txt file)**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#define N 100

int main()

{

    FILE \*fptr;

    int num;

    fptr=fopen("num.txt","w");

    srand(time(0));

    int upper=1000,lower=10;

    for(int i=0;i<N;++i)

    {

        num=i+1;//(rand()%(upper-lower+1))+lower;

        putw(num,fptr);

    }

    fclose(fptr);

    // fptr=fopen("num.txt","r");

    // while((num=getw(fptr))!=EOF)

    // {

    //     printf("%d\n",num);

    // }

    // fclose(fptr);

    return 0;

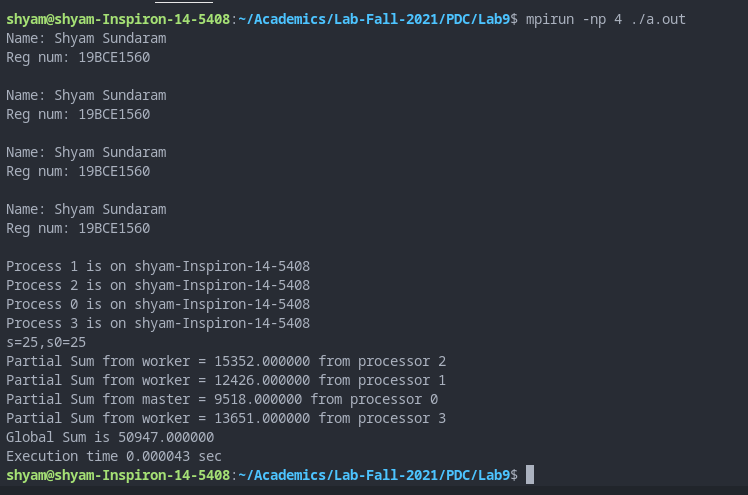
}

**COMMANDS**

mpicc mpisum.c

mpirun -np 4 ./a.out

**OUTPUT**



**OBSERVATION**

Each worker calculated their respective sums and they are added at the end to produce the final sum of all numbers. The sum is verified to be correct.

**CONCLUSION**

We have found the sum of numbers read from a text file using MPI in C.