Parallel Processing Assignment-1

Karthik konyala - R11901409
Pavan Kalyan Anchula-R11904727
Gowtham Edumudi -R11912904
Harshad Hrithik Lanka- R11890176

Main Idea of the program:

The main aim is to achieve an overlap of receiving some rows(0-49 rows) with the computation row sums for other rows (0-49rows).

Non Blocking Communication using MPI_Irecv:

Allowing one processor(processor_1) to receive the data and also doing other tasks such as computation.Using MPI_Irecv will reduce idle time and achieve overlapping of communication with computation.

Using Chunks:

I used chunks of size 5(it can be changed) which helps in splitting the data and send to the processor_1 where it allows a program for more effective overlap communication and computation.When we are receiving one chunk the other chunk can be processed helps in better usage of the computation resources.

MPI_Wait:
Blocking wait, this makes sure that processor_1 doesn't proceed to work on data that has not been fully received.

→After data is generated in processor_0,it computes row sums of rows 50-99 and sends the first 50 rows to process_1.

→Process 1 receives data in chunks, computes row sums for each chunk as soon as it arrives, and sends the final result back to Process 0.

→This approach maximizes energy efficiency by making sure that the process is never idle.

Commands to run the code

mpicc -o assign_1 assign_1.c
Mpirun -np 4 ./assign_1

Ctrl+c to terminate the program.

Output

```
karthikkonyala@Karthiks-Laptop parallel processing % mpirun -np 4 ./assign_1
  328350  328450  328550  328650  328750  328850  328950  329050  329150  329250
  329350  329450  329550  329650  329750  329850  329950  330050  330150  330250
  330350  330450  330550  330650  330750  330850  330950  331050  331150  331250
  331350  331450  331550  331650  331750  331850  331951  332050  332150  332250
  332350  332450  332550  332650  332750  332850  332950  333050  333150  333250
  333350  333450  333550  333650  333750  333850  333950  334050  334150  334250
  334350  334450  334550  334650  334750  334850  334950  335050  335150  335250
  335350  335450  335550  335650  335750  335850  335950  336050  336150  336250
  336350  336450  336550  336650  336750  336850  336950  337050  337150  337250
  337350  337450  337550  337650  337750  337850  337950  338050  338150  338250
 Process 0 Communication time: 0.000079 seconds
 process 1 communication time: 0.000007 seconds
 process 1 Computation time: 0.000014 seconds
```