Nirali Gundecha

Document

Parallelizing Wave Diffusion with MPI and Open MP is hybrid implementation.
Given: Water surface in a two-dimensional square bucket. The water surface can go up and down between 20.0 and -20.0 through the wave dissemination. No wave at time t means Zt_i,j = 0.0.

To simulate wave dissemination over this water surface, I will calculate t=0, t=1 and t=2. Then rotate the matrix for further calculations in fashion t=0 <- t=1 and t=1 <- t=2. I have used static scheduling to parallelize with Open MP.
While parallelizing using MPI, I calculated *stipe* sizes to send to each of the rank for calculation at respective ranks. And received by boundary sharing ranks where Rank0 and Rank=MPI_size-1 will only share single boundary data.

- To calculate stripe size, I have used mathematical logic to divide work as much as possible evenly.
  Calculated Remainder and divided remaining rows to ranks < Remainder.
  Example: for Size=103, 4 Ranks will divide work amongst them
  Rank0 = 25, Rank1= 25, Rank2 = 25, Rank3 = 25
  Remainder = 103/4 =3
  After implementing logic, ranks will get:
  Rank0 = 25+1=26
  Rank1= 25+1=26
  Rank2 = 25+1=26
  Rank3 = 25+0=25
- Calculate First and Last of the stripe
  First is calculated inclusive of the num of row
  Last is calculated exclusive of the num of row
- Using the wave formula given, calculated for t=1 and t=2
- At t=2 to max_time -1, I am sending and receiving data to respective ranks for calculations.
- Rotating matrix to store upcoming data in Z[0] and Z[1]
- Finally, sending calculate data from all other ranks and receiving at Rank0.

Performance Improvement:

(1) The performance improvement with four machines (i.e., four ranks)

```
[nirali09@cssmpi1h prog2]$  mpirun -np 1 ./Wave2D_mpi 588 500 0 1 >out11.txt
Rank[0]'s range = 0 ~ 587
Elapsed time = 7410865
[nirali09@cssmpi1h prog2]$ mpirun -np 6 ./Wave2D_mpi 588 500 0 1 > out61.txt
Rank[0]'s range = 0 ~ 97
Rank[2]'s range = 196 ~ 293
Rank[1]'s range = 98 ~ 195
Rank[5]'s range = 490 ~ 587
Rank[3]'s range = 294 ~ 391
Rank[4]'s range = 392 ~ 489
```

Elapsed time = 3353541
7410865/ 3353541 = 2.209 times

(2) The performance improvement with four machines (i.e., four ranks) with multithreading
[nirali09@cssmpi1h prog2]$ mpirun -np 6 ./Wave2D_mpi 588 500 0 4 > out64.txt
Rank[1]'s range = 98 ~ 195
Rank[0]'s range = 0 ~ 97
Rank[2]'s range = 196 ~ 293
Rank[5]'s range = 490 ~ 587
Rank[4]'s range = 392 ~ 489
Rank[3]'s range = 294 ~ 391
Elapsed time = 2990132
7410865 / 2990132= 2.47 times