Lab 8

Jack Jiang

April 22, 2021

Tasks

1. Setup Coding Environment – Done
2. Review Provided Code
   (a) In the vector structure, the data is distributed by distributing the data amongst the number of processors used.Data is an array of floats and holds values of type float in it when allocated. The data then can be distributed amongst the other processors in use and it is distributed evenly using halo exchanges with size of the overall ghost padding region (padding) and size for the local padded region (N_padded). Data will start being distributed at the starting index of the unpadded local vector (r0) and be stored in the padded index of the vector structure.

   (b) The function inject_unpadded_data_into_Vector(float* data, Vector* v) is responsible for copying data into the unpadded portion of the local vector. It accomplishes this by calculating the padded index by adding the index of the unpadded data and adds the value of the padding of the vector to it and then for each index in the local vector, its data at the padded index is assigned the value of the data array at the unpadded index.

   (c) i. mpirun -np 2 ./test_linear_function 21 3

      ii. After injecting the unpadded data, I expect v to have data corresponding to the inject array with a padding size of $\frac{k-1}{2}$, where K is the window size, as padding on each side of v's data array, numbered from 0.0 to N-1.

      iii. After applying mirror boundary conditions, the data I expect in v is to be, as stated in the function header, in a format such that say the data has a padding of 3, so its —0123—. Then, after the mirror, it'll become 2100123321.

      iv. After sending/receiving the ghost regions, I expect the data in v to have switched values in the ghost regions as the processes send and receive their values to/amongst each other.

v. After applying the average function, I expect the data in v_avg to be the running average of the data computed so far in the execution of the program.

3. Implement Routines

   (a) Done (Code pushed to Git)

4. Questions

   (a) The inject/extract operations can be performed in one line by having an independent pointer pointing to the values you want to inject/extract and injecting/extracting with the pointer.

   (b) The output from each process for the average vector:

```
Average data
————————————
0.3333333433
1.0000000000
2.0000000000
3.0000000000
4.0000000000
5.0000000000
6.0000000000
7.0000000000
8.0000000000
9.0000000000

10.0000000000
11.0000000000
12.0000000000
13.0000000000
14.0000000000
15.0000000000
16.0000000000
17.0000000000
18.0000000000
19.0000000000
19.6666660309
```

   The values at the end of the global vector have a different pattern because the values being averaged at these points are the averages including the mirrored padding.

   (c) I would expect for the 0th entry of the 0th process's unpadded array to be 0.0. The value being averaged is the 0th entry and the mirrored boundary conditions before the 0th entry. I would expect for the 9th entry of the 0th processe's unpadded array to be 9.0. The values to be averaged would be the first 9 values, including the 3 ghost padding values in the front. The output from each process for the average vector:

```
Average data
——————————
40.0000000000
41.0000000000
42.0000000000
43.0000000000
44.0000000000
45.0000000000
46.0000000000
47.0000000000
47.8571434021
48.4285697937
48.7142868042

10.0000000000
11.0000000000
12.0000000000
13.0000000000
14.0000000000
15.0000000000
16.0000000000
17.0000000000
18.0000000000
19.0000000000

20.0000000000
21.0000000000
22.0000000000
23.0000000000
24.0000000000
25.0000000000
26.0000000000
27.0000000000
28.0000000000
29.0000000000

30.0000000000
31.0000000000
32.0000000000
33.0000000000
34.0000000000
35.0000000000
```

```
36.0000000000
37.0000000000
38.0000000000
39.0000000000

1.2857142687
1.5714285374
2.1428570747
3.0000000000
4.0000000000
5.0000000000
6.0000000000
7.0000000000
8.0000000000
9.0000000000
```

(d) All the entries from the average data output have indeed changed as shown below:

```
Average data
____________
0.3333333433
1.6666666269
4.6666665077
9.6666669846
16.6666660309
25.6666660309
36.6666679382
49.6666679382
64.6666641235
81.6666641235

100.6666641235
121.6666641235
144.6666717529
169.6666717529
196.6666717529
225.6666717529
256.6666564941
289.6666564941
324.6666564941
361.6666564941
387.0000000000
```

The values being printed to output from each process after averaging is the running average of the quadratic calculation at each entry.

5. I received no assistance on this assignment. I did assist Shaan Chudasama in the concepts and coding portion, helping him get an idea of how to approach the functions and what each should do.