Assignment: Extra Credit

The purpose of this assignment is for you to learn more about Implementing openmp along with MPI. As usual all time measurements are to be performed on the cluster. Activate OpenMP in MPICXX by passing -fopenmp to the compiler and linker. (Note that if you omit this parameter, the code will probably still compile. But its execution will be sequential.)

1. Priliminary

Run the file "sample.cpp" that uses **MPI and openmp** toverify the output pattern for different number of cores.

2. Reduce Static

Question: Implement computing the sum of an array using static scheduling in MPI. Use any kind of scheduling for OpenMP on different nodes.

Implementation of reduce using static scheduling is in reduce_static.cpp file, and the plots are generated. By analyzing the plots, we can see that for as the processes increase performance of computation increases, however due to communication overhead there is not much change in the performance

3. Reduce Dynamic

Question: Implement computing the sum of an array using dynamic scheduling in MPI. Use any kind of scheduling for OpenMP on different nodes.

Implementation of reduce using dynamic scheduling is in reduce_dynamic.cpp file, and the plots are generated. By analyzing the plots, we can see that for as the processes increase performance of computation increases, however due to communication overhead there is not much change in the performance

4. Compress

Design a compressing algorithm for a random string generated. For example, a string "aaaaabbbbbccccdeee" should output "a5b5c4de3". Use MPI to perform this operation. You can use any kind of scheduling algorithm.

NOTE: If there is only one occurrence of an element, we are not appending a 1 next to it.

Question: Implement computing the sum of an array using the MPI and OpenMP for dynamic processing.

Implementation of compress using openmp is in compression.cpp file, and the plots are generated. By analyzing the plots, we can see that, as the processes increase the performance of computation increases. Speedup is approximately 2 to 3.

Plot speedup charts for all of them.

Observations:

- Including MPI and openmp in the same program did not yield us a very good result irrespective of the scheduling algorithm used.
- Compression of text gave us limited speedups. This is due to the reason that after compression using parallel processes, the main thread has to wait and perform a sequential merge of all the local results which cannot be prevented to maintain the order of occurrences of chars in the string.

This is a joint work of **RAHUL RACHAPALLI (800968032)** and **SANJU GOWDA (800953525).**