<u>**Assignment 2**</u> (Maximum: 80 marks)
Due date: 29-03-2021

Optimize the following collectives based on the current system state and the knowledge of multicore and network topology of the *csews* cluster of CSE. The CSE cluster topology information is specified towards the end of this document. An example of optimization is to combine the collective calls from the MPI ranks of the same node into a single collective call instead of multiple calls from the same node. You may also consider the topology of your allocated nodes to incorporate topology-aware optimizations if you think that may help in real-time. You may additionally include other optimizations depending on the collective function to optimize the below blocking collectives. You will be graded based on the optimizations you perform.

- MPI_Bcast
- MPI_Reduce
- MPI_Gather
- MPI_Alltoallv

You may initialize the elements (doubles) to random values per process, including randomly selected number of elements per rank in case of Alltoallv. Time only the collective call (default and optimized). Your source code (src.c) should include all the above collectives (both default and optimized). Each of these should be invoked for 5 times. Compare the performance (average time for 1 call) of the default with your optimized collective for all the above collective functions.

```
for i in 1 .. 5
   call MPI_xxx_default; call MPI_xxx_optimized;
```

Report results for the following configurations.

```
for execution in 1 to 10 // (keep this as the outermost loop, helps to know the variability better)
 for P (number of nodes) in 4, 16
  for ppn (number of processes/cores per node) in 1, 8
   for D (doubles) in 16, 256, 2048 (KB)
    mpirun –np P –f hostfile ./code D
```

Plot the time (in seconds) for each data size per node count per core count for every collective function. Use barcharts with error bars (from the 10 executions) for every data point in a plot. Plot the data corresponding to a collective in a separate file, i.e. submit 4 plot files. Plot the times for each P for each ppn for each D in the same plot file. Time in seconds (y-axis) and D (x-axis). An example plot file has been provided.

<u>Execution and submission instructions</u>

Create 'Assignment2' directory on git. It should necessarily contain the source code ('src.c'), 'Makefile', 'readme.pdf', job script ('run.py' or 'run.sh'), plot script and plots. The job script should compile your code using the "make" command. The job script should run all the configurations as specified above. I should be able to run the job script to execute your code (all configurations). The job script execution must generate the output data* files. Name your plots 'plot_X.jpg' for X in {Bcast, Reduce, Gather, Alltoallv}. The 'readme.pdf' should contain an understandable explanation of your code, what optimizations did you perform and why, your observations regarding these optimizations and the performance from the plots, the 4 plots, any additional testing that you did for better understanding and any issues that you may have faced with the code, experimental setup etc. You must use a script that generates machinefile/hostfile on-the-fly based on the node status so that your jobs never fail. You may peruse the script provided on Piazza to create a list of nodes from different network groups to conduct experiments. The job script must generate the hostfile before the runs.

Suggestions

- Follow the above instructions carefully. You will be penalized otherwise. For example, missing “make”, incorrect source code file name, missing graphs, etc.
- Use git.cse.iitk.ac.in as a real git repo. We will monitor your progress.
- Document your code neatly (ensure readability and understandability). We will award you marks for this.
- Note that your runtime may be affected if you have too many output lines for debugging.

Final submission

- Send email to {pmalakar, avikpal, lavleshm, piprotar, mabir, samvid, tusharag, vcvaibh}@cse.iitk.ac.in
- Do not send any attachments (-5 penalty for those who do).
- Include the following in your email (only 1 email, please!).
    - Names and roll numbers of group members
    - Git repo link
    - Number of early/late days

A sample script to generate hostfile containing nodes from different groups and a sample plot script has been uploaded on Piazza (helper_scripts.tar.gz). You will need to modify the plot script to include code to read data from your output files.

CSE node groups (6)
Intra-group distance (i.e. #hops within a group) = 2
Inter-group distance (i.e. #hops across groups) = 4

csews1, csews2, csews3, csews4, csews5, csews6, csews7, csews8, csews9, csews10, csews11, csews12, csews14, csews15, csews16, csews31.

csews13, csews17, csews18, csews19, csews20, csews21, csews22, csews23, csews24, csews25, csews26, csews27, csews28, csews29, csews30. csews32.

csews33, csews34, csews35, csews36, csews37, csews38, csews39, csews40, csews41, csews42, csews43, csews44, csews46.

csews45, csews47, csews48, csews49, csews50, csews51, csews52, csews53, csews54, csews56, csews58, csews59, csews60, csews61.

csews62, csews63, csews64, csews65, csews66, csews67, csews68, csews69, csews70, csews71, csews72, csews73, csews74, csews75, csews76, csews77, csews78.

csews79, csews80, csews81, csews82, csews83, csews84, csews85, csews86, csews87, csews88, csews89, csews90, csews91, csews92.