Supercomputing in Plain English
Exercise #7: MPI Collective Communications

In this exercise, we'll use the same conventions and commands as in Exercises #1, #2, #3, #4, #5 and #6. You should refer back to the Exercise #1, #2, #3, #4, #5 and #6 descriptions for details on various Unix commands.

In the exercise, you'll again be parallelizing and benchmarking, but this time you'll be parallelizing with MPI collective communications instead of with MPI point to point communications. Then you'll benchmark various numbers of MPI processes using various compilers and various levels of compiler optimization.

Here are the steps for this exercise:

1. Log in to the Linux cluster supercomputer (`sooner.oscer.ou.edu`).
2. Choose which language you want to use (C or Fortran90), and `cd` into the appropriate directory:

   `% cd ~/SIPE2011_exercises/NBody/C/`

   OR:

   `% cd ~/SIPE2011_exercises/NBody/Fortran90/`
3. Copy the `Serial` directory to a new `MPI` directory:

   `% cp -r MPI_p2p/ MPI_collective/`
4. Go into your `MPI` collective communications directory:

   `% cd MPI_collective`
5. Parallelize the code using MPI collective communication, specifically replacing your various `MPI_Send` and `MPI_Recv` routine calls with one or more collective communication routine calls.
6. Set the `MPI_COMPILER` and `MPI_INTERCONNECT` environment variables.
7. Compile using `make`. You may need to do this multiple times, debugging as you go.
8. Submit the batch job and let it run to completion. If it seems to take a very long time, probably you have a bug.
9. For each run, once the batch job completes:
   a. Examine the various output files to see the timings for your runs with executables created by the various compilers under the various levels of optimization.
   b. Profile, as described above.
10. Continue to debug and run until you've got a working version of the code.