1 Scalability: strong and weak scaling

When using HPC clusters, it is almost always worthwhile to measure the parallel scaling of your jobs. The measurement of strong scaling is done by testing how the overall computational time of the job scales with the number of processing elements (being either threads or MPI processes), while the test for weak scaling is done by increasing both the job size and the number of processing elements. The results from the parallel scaling tests will provide a good indication of the amount of resources to request for the size of the particular job.

1. Copy the C codes under the Practical folder of Week 2 from Brightspace/Canvas: `benchmark.h`, `benchmark_mpi.c`, and `benchmark_openmp.c` to your sciprog account.

    - You can see here that we vary the job size 10^6, 10^7, 10^8, and 10^9 and number of processing elements (OpenMP threads and MPI processes). The job sizes are set by the "size" variables and the processing elements by the upper bound of the for loop for the OpenMP code and by the -n flag with mpirun for MPI.
    - Add print statements in main to observe the change in time taken as you increase the job size and the number of processing elements.

2. Fortran codes: inc_serial.f90, inc_omp.f90 and inc_mpi.f90

3. sciprog is only a 2-core machine, so we have provided results from a 100-core machine for both OpenMP and MPI. Note that the -1 at the end of the OpenMP results corresponds to a serial run.

    - Calculate the relative and absolute efficiency for the given lengths and processes above. Plot them in your plotting language of choice. Observe Amdahl's law and comment on any trends you see in the plots.
    - Calculate the relative and absolute speedup for the given lengths and processes above. Again plot them in your plotting language of choice. Observe Gustafson-Barsis' law and comment on any trends you see in the plots.