



# Parallel IO Performance on ARCHER2

David Henty

## 1 Introduction

The aim is to investigate how parallel IO performance is affected by a number of factors:

- the number or processes or nodes performing serial IO;
- choice of striping settings for parallel IO;
- data aggregation within the Cray MPI-IO library;
- the choice of MPI-IO, HDF5, NetCDF or ADIOS2.

Running the exercises should also give you some feeling for the kinds of IO rates achievable in practice, enabling you more easily to judge whether the IO phases of your own codes can be improved.

## 2 Test code

The test code is called `benchio` and is a simple Fortran program that performs parallel IO using a number of serial and parallel approaches. It is written in Fortran for purely historical reasons: the IO libraries it uses are exactly the same as would be used by a C or C++ program, so performance results should be equally applicable to all languages.

For details, see `README.md` at <https://github.com/davidhenty/benchio/>.

## 3 Setup

On ARCHER2 **you must work in the /work/ file system**, i.e. `cd /work/project/project/username/`.

Obtain the code from github:

```
git clone https://github.com/davidhenty/benchio
```

Before compiling, you must load some additional modules:

```
module use /work/z19/shared/sfarr/modulefiles
module load adios/2.8.3
module load cray-hdf5-parallel
module load cray-netcdf-hdf5parallel
```

Now you should be able to compile:

```
make -f Makefile-archer2
```

Before running the code, you should configure the three output directories as follows:

- remove any striping on unstriped (i.e. set it to 1): `lfs setstripe -c 1 unstriped`
- striped should be set to a stripe count of 2: `lfs setstripe -c 2 striped`
- fullstriped should have the maximum (i.e. system-selected) value: `lfs setstripe -c -1 fullstriped`

## 4 Initial exercise

You should be able to submit a job using the supplied Slurm script:

```
sbatch archer2nov23.job
```

This will run a very simple test with serial IO of a 1 GiB file from a single process (see the github page for details of what the benchmark does and the command-line options):

```
./benchio 512 512 512 local serial unstriped
```

Note that this Slurm script will automatically submit to today's reserved queue – after the course ends you can use `archer2.job` to run in the short queue.

What IO rates do you see from a single process?

## 5 Basic IO performance

We can try and estimate the maximum IO bandwidth from a node (128 CPU-cores) using the following settings:

```
#SBATCH --tasks-per-node=128
...
./benchio 256 256 256 local proc unstriped
```

It is a good idea to reduce the local array sizes as the total amount of data can become very large when running on hundreds of processes.

We can see if the limiting factor is the speed of the disks or the bandwidth from a single node by running on multiple nodes:

```
#SBATCH --nodes=2
#SBATCH --tasks-per-node=128
...
./benchio 256 256 256 local proc unstriped
```

How do the bandwidths from these tests compare?

Other things you could look at:

- To test if the MDS is a limiting factor you could write the same amount of data but using fewer processes, e.g. a  $512^3$  array with only 16 tasks per node.
- For serial IO we would **not** expect striping to give significant benefits. What happens if you re-run the tests using the `striped` directory?
- Do the results look any different if you do IO to the NVMe (solid-state storage) filesystem at `/scratch-nvme/project/project/username/`?

## 6 MPI-IO performance

For parallel IO we would hope that the bandwidth scales with the number of stripes.

Try the following tests:

```
#SBATCH --nodes=1
#SBATCH --tasks-per-node=128
...
./benchio 128 128 128 local mpio
```

which performs MPI-IO to all three directories. Now try 2, 4 and 8 nodes - what IO rates do you see?

Other things you could look at:

- We expect that using non-collective IO routines will be very slow. Edit the file `mpio.f90` and replace `MPI_File_write_all` with `MPI_File_write`.  
Note that the performance could be so slow that it is hard to measure (e.g. it takes too long to run). If so, try running on smaller numbers of processes.
- How are the IO rates affected if you do not use the native data format, e.g. you replace this with `external32`?

## 7 Configuring the Lustre filesystem

Although by far the most important factors for achieving good parallel IO rates are to use collective IO and stripe the file across multiple OSTs, there are other settings that can be changed.

- The Cray MPI-IO library can output useful statistics on how it performed your IO – to get this debug output, set the following variable in your Slurm script: `export MPICH_MPIIO_STATS=1`
- Does stripe size have any effect, e.g. what if you use 4 MiB stripes instead of the default 1 MiB?
- Try using the alternative implementation of the Cray MPI library by switching modules in your Slurm script (you do not need to recompile) - add these lines *after* the other module commands:  

```
module swap craype-network-ofi craype-network-ucx
module swap cray-mpich cray-mpich-ucx
module load libfabric
```
- By default, the MPI-IO library only allocates a single IO process for each stripe. This may not be optimal - you can try increasing this number:  
`export MPICH_MPIIO_HINTS=*:cray_cb_nodes_multiplier=2,*:cray_cb_write_lock_mode=2`

The first option doubles the number aggregators compared to the default (i.e. here we would get two aggregators per stripe) and the second option stops these aggregators from locking access to the stripes. Using \* applies this to all files.

## 8 Alternative IO libraries

Repeat some of the MPI-IO experiments using a higher-level library like HDF5. To run on a single node:

```
#SBATCH --nodes=1
#SBATCH --tasks-per-node=128
...
./benchio 128 128 128 local hdf5
```

Is performance similar to MPI-IO? Do the MPI-IO stats indicate data aggregation is used efficiently?

### 8.1 ADIOS2 configuration

By default, ADIOS2 uses one aggregator per node (**not** per stripe as for MPI-IO). You can increase this by editing the `adios2.xml` file.

- does having more than one aggregator per node improve performance?
- remember ADIOS2 writes multiple files under the hood: does file striping affect the IO rates?