How to improve performance with Parallel HDF5

Tuning parallel HDF5 for a specific application on a specific system requires playing with a lot of tunable parameters many of which are specific to certain platforms. Not all hints are applicable to all platforms, and some hints may be ignored even if they can be applied. The best practice here is to look at each system's webpage on how to tune I/O parameters. For example, Hopper, a Cray XE6 supercomputer at NERSC, has a webpage specifically on how to tune parallel I/O parameters for specific file systems:

> http://www.nersc.gov/users/computational-systems/hopper/file-storage-and-i-o/

Here are some general parameters that users should consider tuning when they see slow I/O performance from HDF5:

HDF5 parameters:

1. Chunk size and dimensions: If the application is using chunked dataset storage, performance usually varies depending on the chunk size and how the chunks are aligned with block boundaries of the underlying parallel filesystem. Extra care must be taken on how the application is accessing the data to be able to set the chunk dimensions.
2. Metadata cache: it is usually a good idea to increase the metadata cache size if possible to avoid small writes to the file system. See: H5P_SET_MDC_CONFIG
3. Alignment properties: For MPI IO and other parallel systems, choose an alignment which is a multiple of the disk block size. See: H5P_SET_ALIGNMENT

MPI-IO parameters:

There are several MPI-I/O parameters to tune. Usually it is done by setting info keys in the info object passed to HDF5. Some implementations might allow other ways to pass those hints to the MPI library. The MPI standard reserves some key values. An implementation is not required to interpret these key values, but if it does interpret the key value, it must provide the functionality described. The best thing to do again here is to consult with the specific MPI implementation and system used documentation to see what parameters are available to tune. For example, ROMIO in MPICH provides a user guide with a section describing the hints that are available to tune:

> http://www.mcs.anl.gov/research/projects/romio/doc/users-guide.pdf

Here are some general parameters that are usually tunable:

1. `cb_block_size` (integer): This hint specifies the block size to be used for collective buffering file access. Target nodes access data in chunks of this size. The chunks are distributed among target nodes in a round-robin (CYCLIC) pattern.
2. `cb_buffer_size` (integer): This hint specifies the total buffer space that can be used for collective buffering on each target node, usually a multiple of cb_block_size.
3. `cb_nodes` (integer): This hint specifies the number of target nodes to be used for collective buffering.

MPI implementations other than ROMIO might provide a way to tune those parameters, but not necessarily through info hints. OMPIO (an Open MPI native MPI-IO implementation) for example uses OMPI MCA parameters to tune those hints.

Parallel File System parameters:

Depending on the parallel file system and what version it is, there are several ways to tune performance. It is very hard to come up with a general list of tunable parameters for all file systems, since there are not many common ones. Users should individually check the documentation for the particular file system they are using.

For most parallel file systems the two parameters that are usually tunable and very important to consider are:

1. Stripe size: Controls the striping unit (in bytes).
2. Stripe Count: Controls the number of I/O devices to stripe across.

For Blue Gene /P and /Q, one can set the environment variable `BGLOCKLESSMPIO_F_TYPE` to `0x47504653` (the GPFS file system magic number). ROMIO will then pretend GPFS is like PVFS and not issue any `fcntl()` lock commands.

Some IBM specific hints:

> http://www-01.ibm.com/support/knowledgecenter/SSFK3V_1.3.0/com.ibm.cluster.pe.v1r3.pe500.doc/am107_ifopen.htm?lang=en

Some Cray specific hints:

> https://fs.hlrs.de/projects/craydoc/docs/man/xe_mptm/51/cat3/intro_mpi.3.html