HDF5 Troubleshooting

Gerd Heber, The HDF Group

A Simple Problem

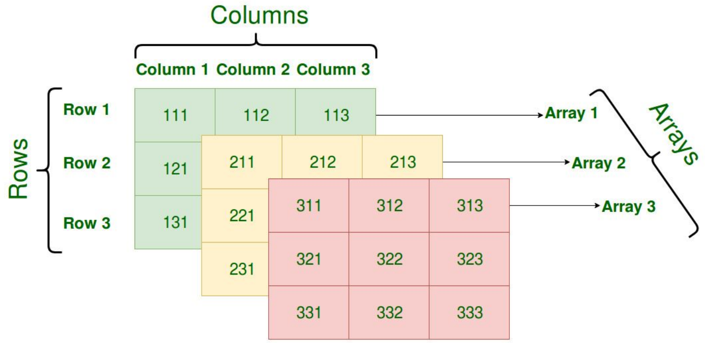
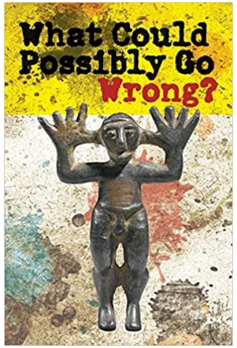


Figure: GeeksForGeeks

Writing multiple 2D array variables over time:

```
ACROSS P processes arranged in a R x C process grid
    FOREACH step 1 .. S
        FOREACH count 1 .. A
            CREATE a double ARRAY of size [X,Y] | [R*X,C*Y] (strong | weak)
            (WRITE | READ) the ARRAY (to | from) an HDF5 file
        END
    END
END
```

S(teps) = 20, A(rrays) = 500, X = 100, Y = 200 (See adios_iotest)



Missing Information

- How are the array variables represented in HDF5?
  - 2D, 3D, 4D datasets
  - Are the extents known a priori?
  - How are the dimensions ordered?
  - Groups?
- How (order) is the data written and is the data read the same way?
- What’s that storage layout?
  - How many physical files?
  - Contiguous or chunked, etc.
  - Is the data compressible?
- What’s the file system or data store?
- Collective vs. independent MPI-IO
- ...

Basic Combinations (24)

- Six griddings

/step=[0..10]/array=[0..499]	Dataset {100, 200}
/array=[0..499]/step=[0..19]	Dataset {100, 200}
/step=[0..20]	Dataset {500, 100, 200}
/array=[0..499]	Dataset {20, 100, 200}
/dataset	Dataset {20, 500, 100, 200}
/dataset	Dataset {500, 20, 100, 200}

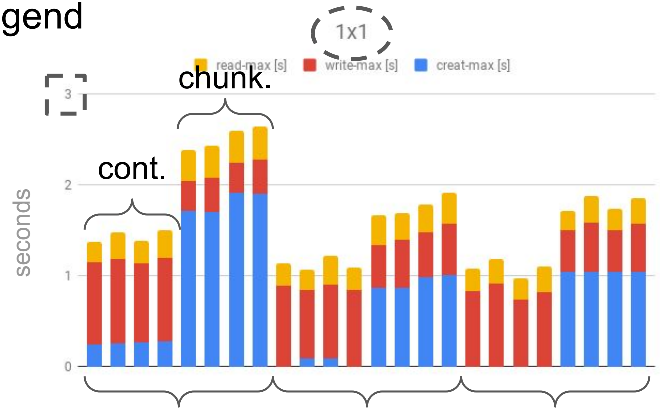
- Two layouts
  - Contiguous or chunked
- Two MPI modes
  - Collective or independent
- ...

Baseline

- Run **24 parameter configurations**
- Weak scaling
  - Each process writes 500*100*200*8 (~ 80 MB) per step (20 steps)
- Single processor, 4 processor grids: 1 x 4, 2 x 2, 4 x 1
- Measure times for dataset creation, write, and read

Taller bar “=” bad (longer time)

Legend



chunk.

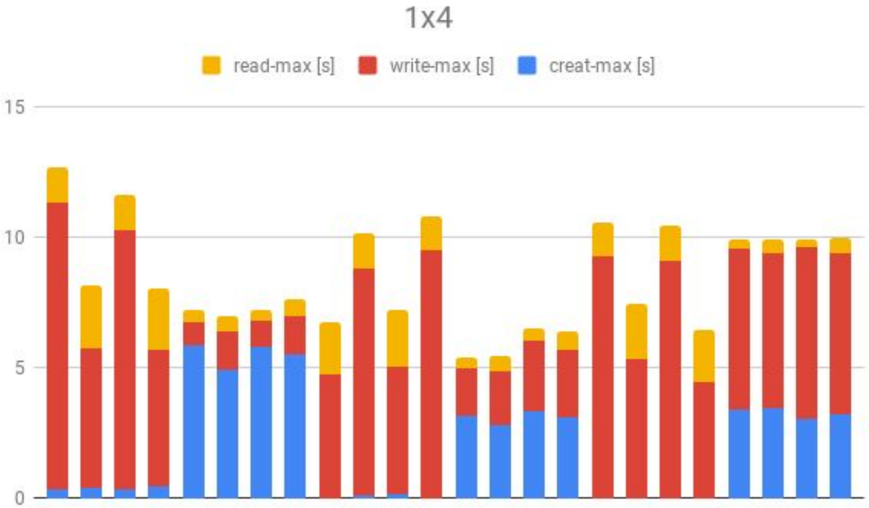
cont.

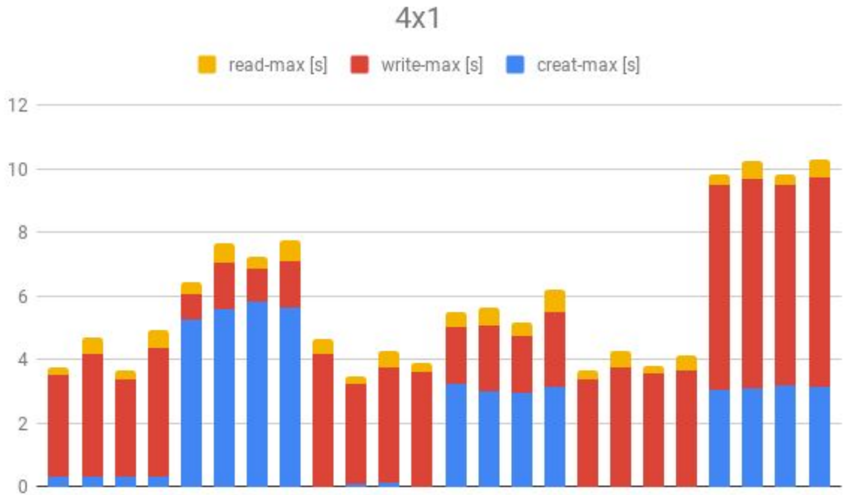
dim 2

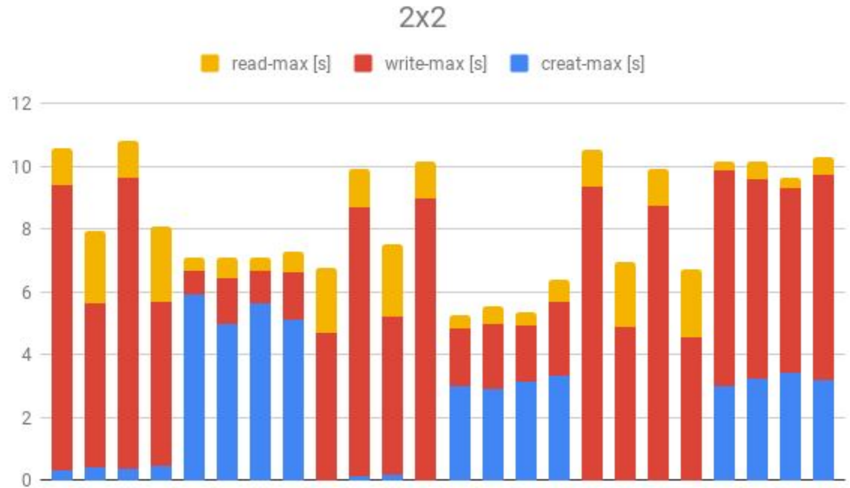
dim 3

dim 4

Shorter bar “=” good (shorter time)

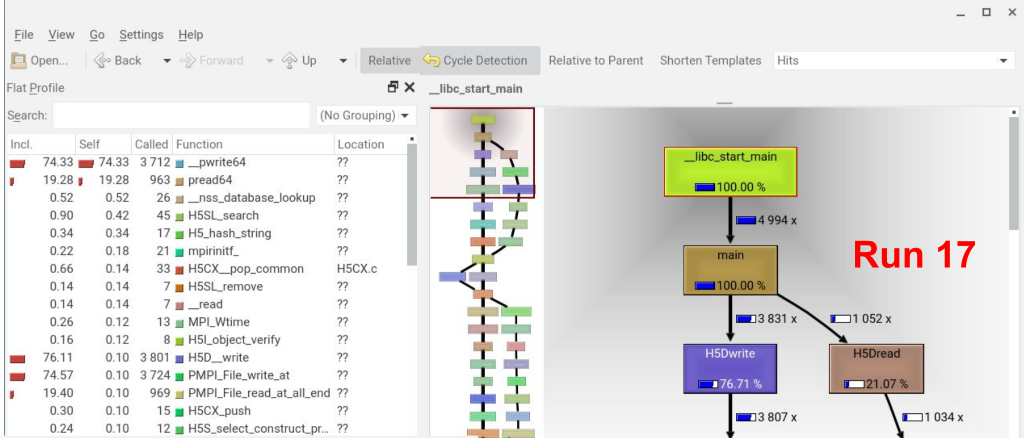






Gperftools

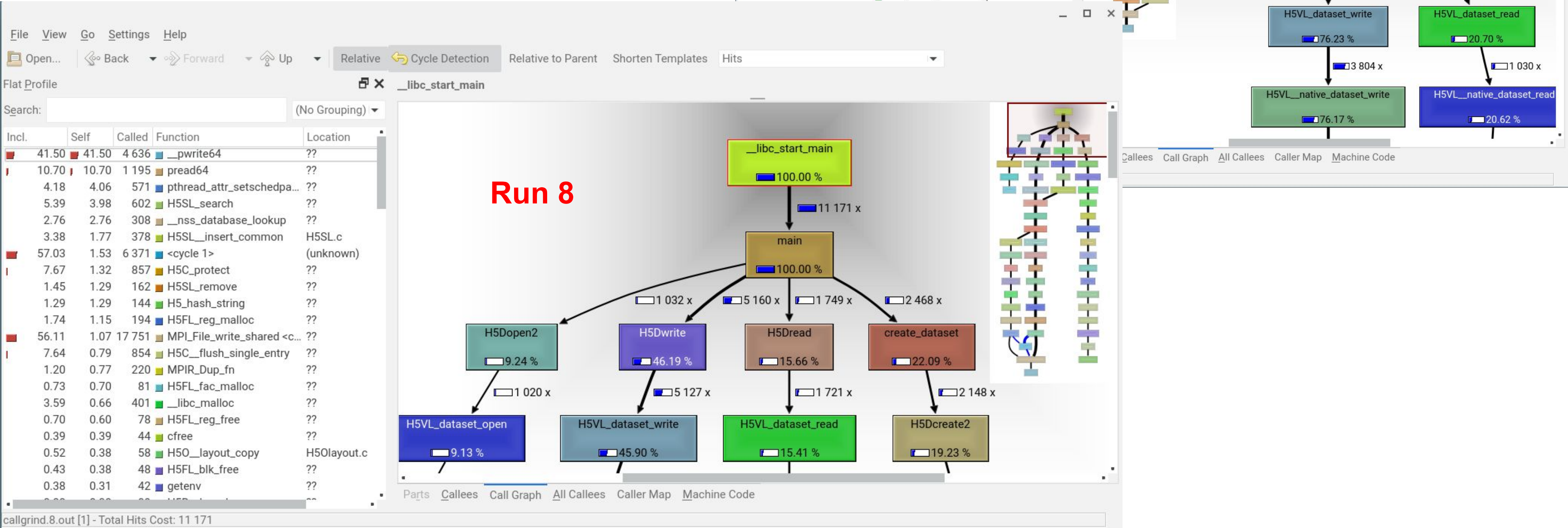
- Source on GitHub
- Use w/ KCachegrind
- See `profiler.sh`


File View Go Settings Help
Open... Back Forward Up
Relative Cycle Detection Relative to Parent Shorten Templates Hits
Flat Profile
__libc_start_main
__libc_start_main 100.00 %
4 994 x
main 100.00 %
3 831 x
1 052 x
H5Dwrite 76.71 %
H5Dread 21.07 %
3 807 x
1 034 x
H5VL_dataset_write 76.23 %
H5VL_dataset_read 20.70 %
3 804 x
1 030 x
H5VL__native_dataset_write 76.17 %
H5VL__native_dataset_read 20.62 %
Callees Call Graph All Callees Caller Map Machine Code

Run 17

Search: (No Grouping)

Incl.	Self	Called	Function	Location
74.33	74.33	3 712	__pwrite64	??
19.28	19.28	963	pread64	??
0.52	0.52	26	__nss_database_lookup	??
0.90	0.42	45	H5SL_search	??
0.34	0.34	17	H5_hash_string	??
0.22	0.18	21	mpirinitf_	??
0.66	0.14	33	H5CX__pop_common	H5CX.c
0.14	0.14	7	H5SL_remove	??
0.14	0.14	7	__read	??
0.26	0.12	13	MPI_Wtime	??
0.16	0.12	8	H5I_object_verify	??
76.11	0.10	3 801	H5D__write	??
74.57	0.10	3 724	PMPI_File_write_at	??
19.40	0.10	969	PMPI_File_read_at_all_end	??
0.30	0.10	15	H5CX_push	??
0.24	0.10	12	H5S_select_construct_pr...	??



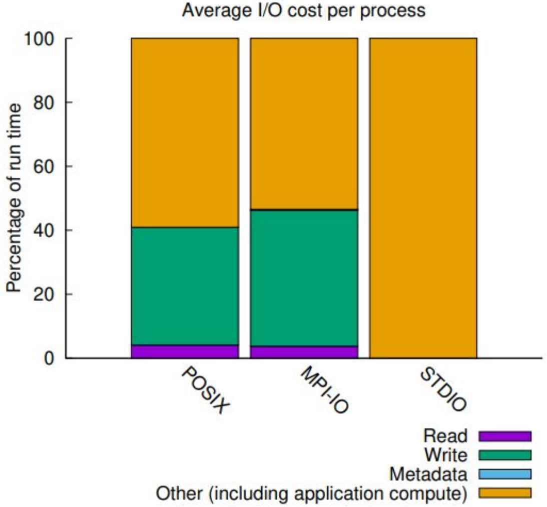
Run 8

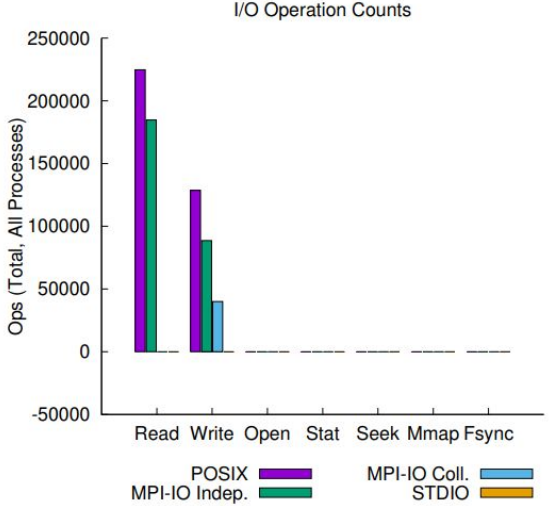
Search: (No Grouping)

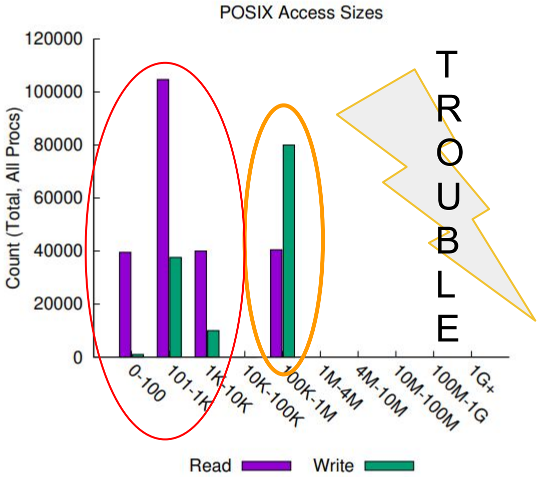
Incl.	Self	Called	Function	Location
41.50	41.50	4 636	__pwrite64	??
10.70	10.70	1 195	pread64	??
4.18	4.06	571	pthread_attr_setschedpa...	??
5.39	3.98	602	H5SL_search	??
2.76	2.76	308	__nss_database_lookup	??
3.38	1.77	378	H5SL__insert_common	H5SL.c
57.03	1.53	6 371	<cycle 1>	(unknown)
7.67	1.32	857	H5C_protect	??
1.45	1.29	162	H5SL_remove	??
1.29	1.29	144	H5_hash_string	??
1.74	1.15	194	H5FL_reg_malloc	??
56.11	1.07	17 751	MPI_File_write_shared <c...	??
7.64	0.79	854	H5C__flush_single_entry	??
1.20	0.77	220	MPIR_Dup_fn	??
0.73	0.70	81	H5FL_fac_malloc	??
3.59	0.66	401	__libc_malloc	??
0.70	0.60	78	H5FL_reg_free	??
0.39	0.39	44	cfree	??
0.52	0.38	58	H5O__layout_copy	H5Olayout.c
0.43	0.38	48	H5FL_blk_free	??
0.38	0.31	42	getenv	??

callgrind.8.out [1] - Total Hits Cost: 11 171

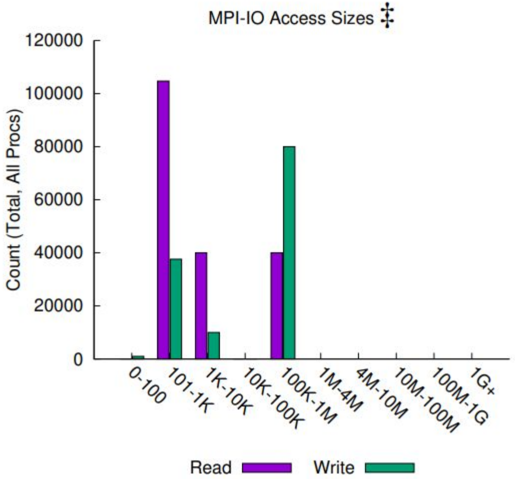
4x1 Run 7


Average I/O cost per process


I/O Operation Counts


POSIX Access Sizes

TROUBLE


MPI-IO Access Sizes ‡