Single-threaded HDF5 application with Bypass VOL connector and internal thread pool

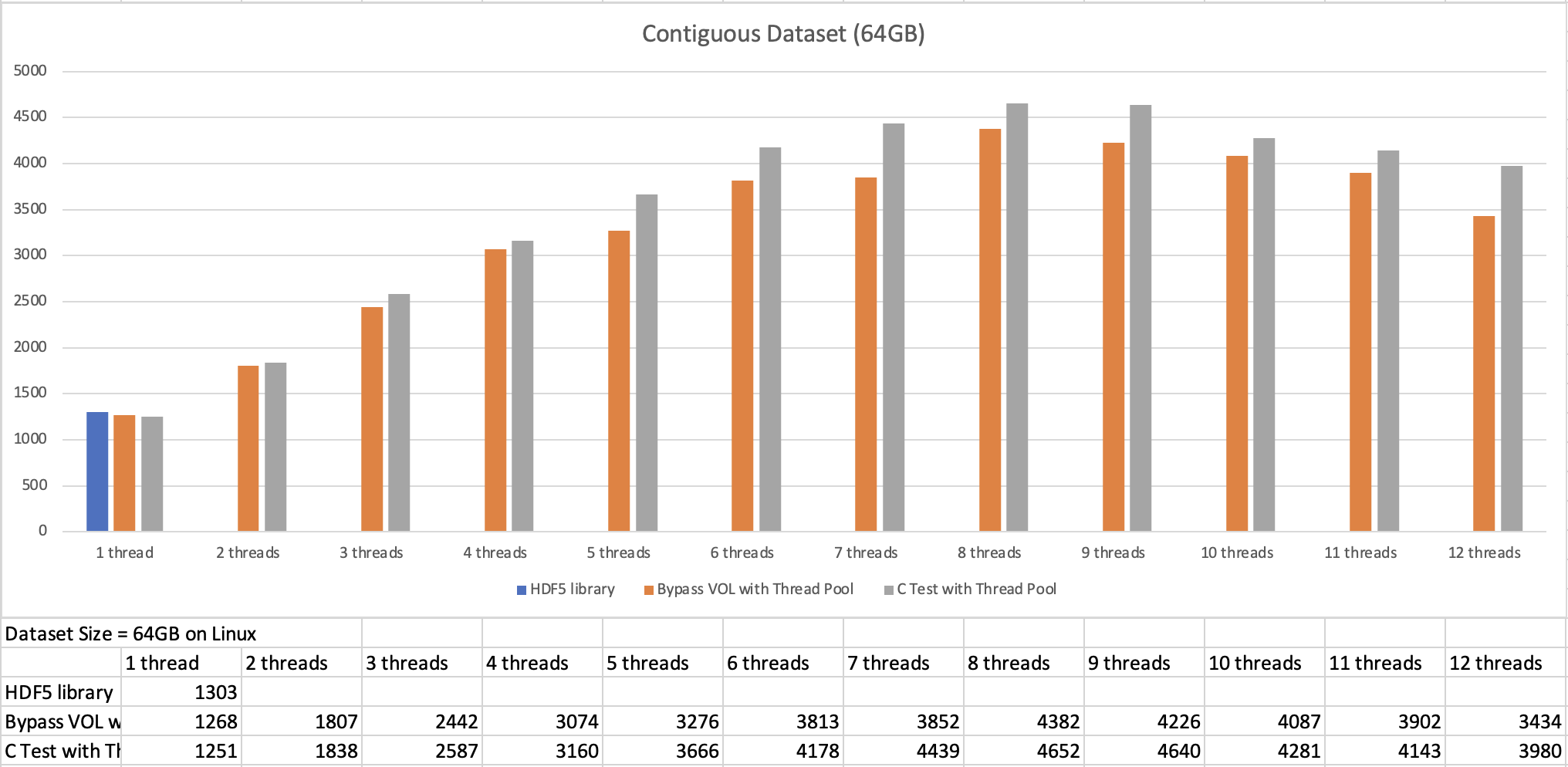
Reading 64GB contiguous and chunked dataset by 4MB hyperslabs.

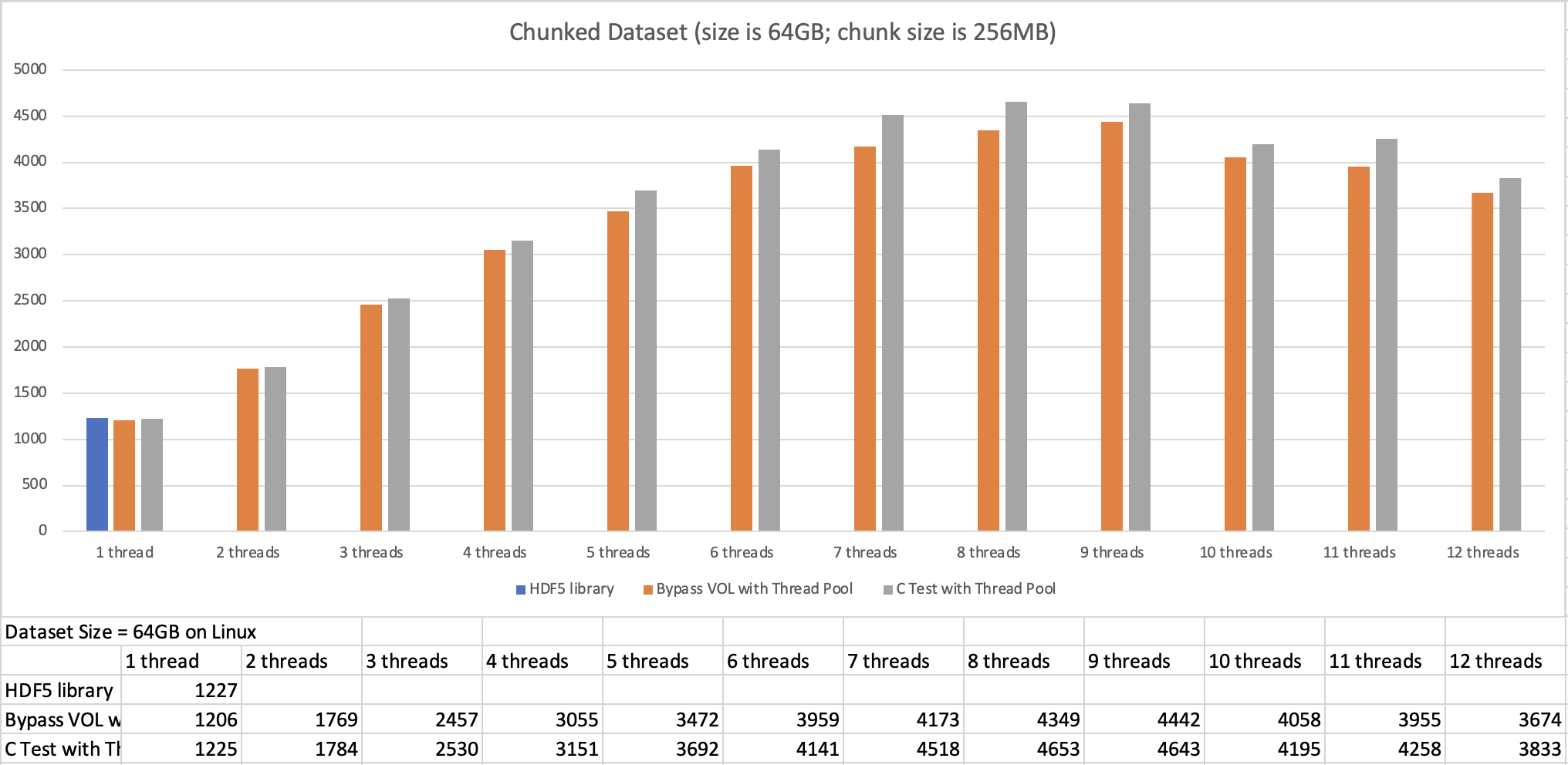
1) Blue bar is a single-threaded HDF5 program.

2) Orange bar is the same HDF5 program; it uses Bypass VOL connector with the thread pool.

3) Grey bar is a C program that mimics the I/O pattern of the Bypass VOL connector. The pattern is stored in the text file as offset and length pairs. The C program also uses the thread pool.

2) and 3) use the thread pool of 1 to 12 threads (horizontal axis). Vertical axis is reading speed in MB/sec.





Multi-threaded HDF5 application with Bypass VOL connector and internal thread pool

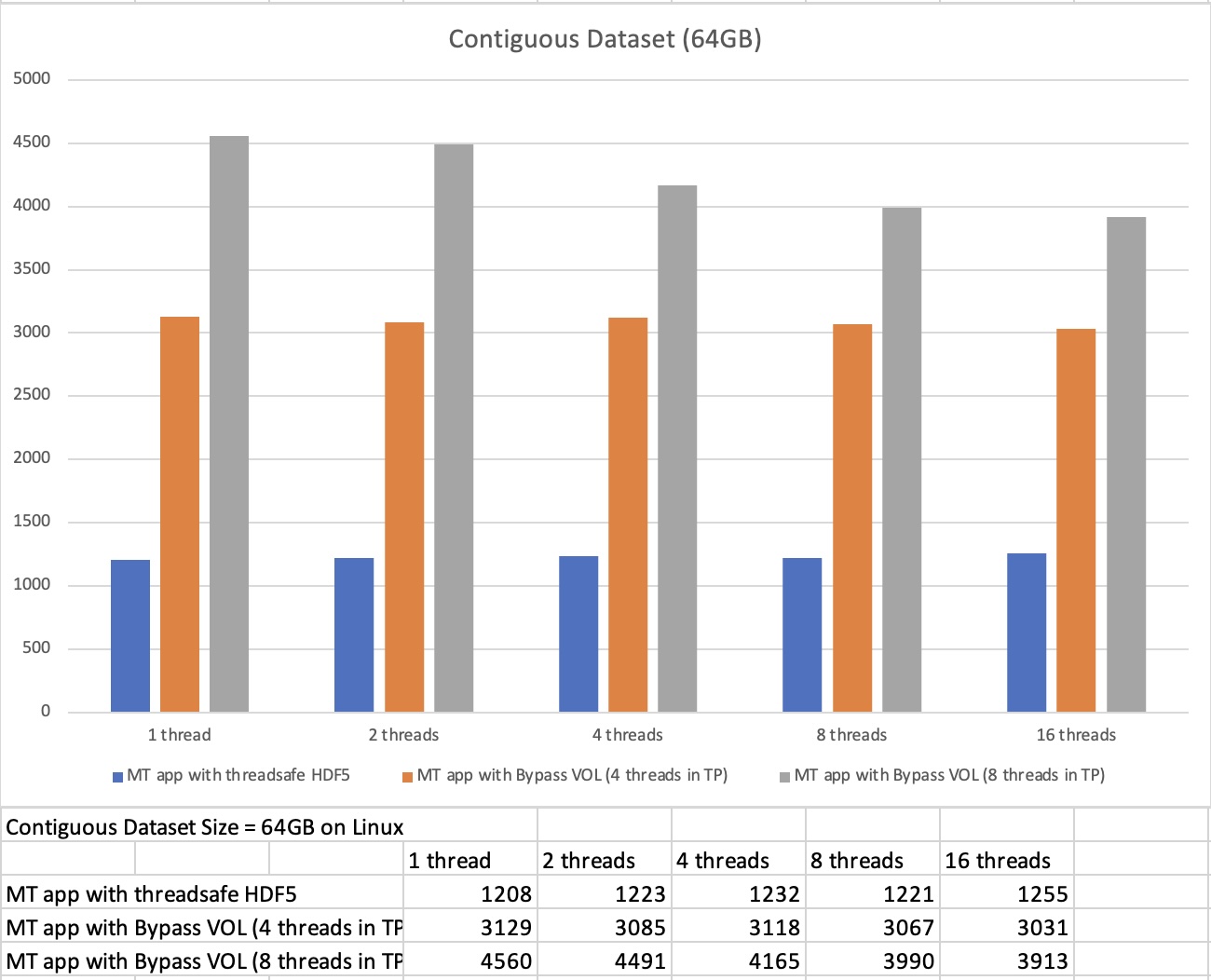
Reading 64GB contiguous and chunked dataset by 4MB hyperslabs

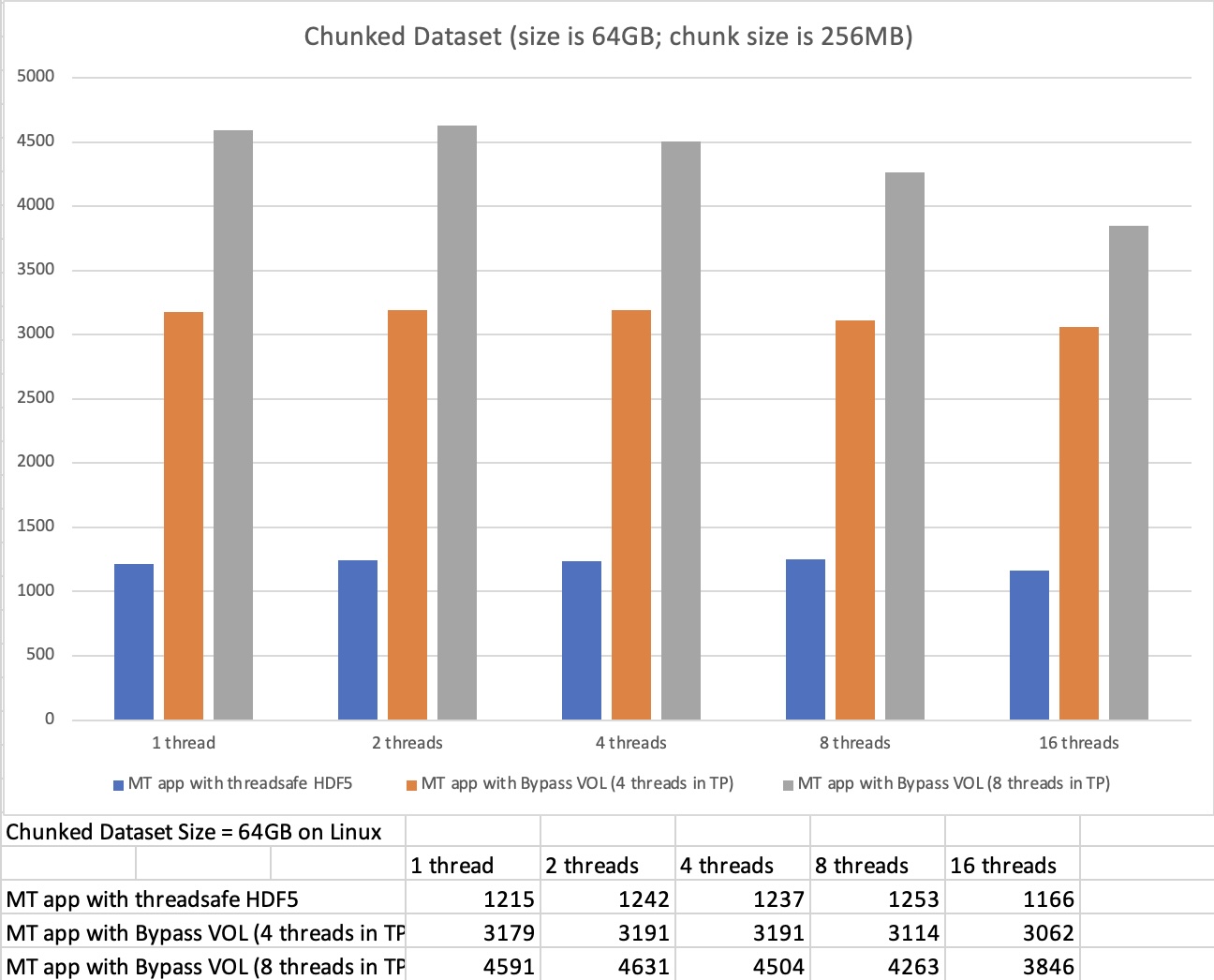
1) Blue bar is a multi-threaded HDF5 application with thread-safe HDF5.

2) Orange bar is the same HDF5 program with Bypass VOL connector; 4 threads are in the thread pool (TP).

3) Grey bar is the same HDF5 program with Bypass VOL connector; 8 threads are in the thread pool (TP).

Horizontal axis is number of threads used by the HDF5 app. Vertical axis is reading speed in MB/sec.





**Summary**

In both cases (single-threaded HDF5 application with the thread pool and multi-threaded HDF5 application with the thread pool) we see nice performance improvement (~ 3x to 4.5x speedup). We need to investigate performance drop in both cases when the number of total threads increases. We are currently working on a benchmark that doesn’t use thread pool.