I/O Performance on the Connection Machine DataVault System

John Krystynak*
Computer Sciences Corporation
NASA Ames Research Center
Moffett Field, CA 94035

Abstract

This paper presents performance measurements of the Connection Machine DataVault system. The DataVault is an I/O system for mass storage. Theoretically, the DataVault configuration at NASA Ames Numerical Aerodynamic Simulation Facility is capable of transfer rates of 32 megabytes/sec. The major limitation of the current DataVault configuration is its inability to use more than one CMIO bus to communicate between the Connection Machine and the DataVault. Overall however, the DataVault is an effective I/O system. Performance statistics show that actual DataVault I/O performance can be close to peak theoretical rates. I/O system behavior from CM Fortran and C/PARIS is described. Graphs of performance results are given, with interpretation focusing on how to realize good CM I/O system throughput.

1 Introduction

The Connection Machine (CM) I/O system is comprised of the CM, the CM I/O busses and the DataVault. This system affects computational fluid dynamics (CFD) applications on the CM since it is where data, grid and solution files are often stored. Because CFD applications and other applications depend on the CM I/O system for mass storage, the performance of the CM I/O system affects overall throughput of many codes. Therefore, it is important to have an idea of I/O performance on the CM system.

Many factors influence DataVault I/O throughput. Some of these factors include: hardware architecture of the system, software interface used, user load on the CM front-ends, and how the system is used. In this paper, the examination is focused on the hardware performance capabilities of the DataVault system. Basic tests are run on the DataVault to determine the maximum performance given a fixed hardware configuration. No attempt is made to simulate the I/O requirements of a CFD application.

The results presented in this paper are intended to verify and explain the behavior of the CM, DataVault and the CM I/O busses. The results are relevant to application developers for predicting and checking I/O performance. Other I/O hardware, such as the CM-HiPPI is similarly affected by the performance of the I/O busses and CM, and some tests may be helpful to users of such hardware.

The key sections of the investigation are:

- Explanation of the CM I/O and DataVault system configuration.
- Explanation of the performance tests.

*Work Supported by NASA Contract NAS 2-12961

- Performance results and graphs.
- Conclusions and summary.

2 CM I/O Configuration at NASA Ames

The layout of the CM I/O system determines the maximum I/O throughput which the CM and DataVault can attain. The key components of the CM I/O system at the Numerical Aerodynamic Simulation (NAS) Facility are as follows:

- 32,768 processor CM-2.
- 25 gigabyte DataVault.
- 2 32MB/sec CM I/O busses.
- 4 CM I/O Controllers (CMIOC).

The CM has four ports (CMIOCs) to the I/O busses, one for each sequencer. Each I/O bus is attached to two ports. At NAS, the even-numbered sequencers are attached to CMIO bus #2, the odd sequencers are attached to CMIO bus #1. The CMIO busses each support a maximum transfer rate of 32MB/sec. The DataVault, however, can only interact with one CMIO bus at a time. This means that the DataVault is limited to a maximum transfer rate of 32MB/sec. A device which can interact with both busses would be capable of 64MB/sec. A *striped* I/O system consisting of 2 or more DataVaults could use both busses for peak transfer rates of 64MB/sec. A CM-HIPPI network device can also use more than one CMIO bus at a time. The architecture of the current CM I/O system is illustrated in figure 1.

3 How the Performance Tests are Conducted

The purpose of the performance tests is to determine I/O performance characteristics of the CM I/O system. The tests are not designed to simulate production codes.

There are two types of performance tests in this paper, slicewise (written in CM Fortran) and fieldwise (in C/PARIS and CM Fortran). Slicewise programs view the CM architecture as a collection of floating point units, each served by 32 CM processors. In slicewise execution, one 32-bit floating point number can be supplied to a floating point unit every clock cycle. A floating point unit takes a 'slice' across 32 CM processors to form an operand each cycle. In fieldwise mode, a floating point unit is supplied with 32 operands from 32 CM processors every 32 cycles, with each CM processor delivering one bit per cycle. The term 'fieldwise' is derived from the notion that each CM processor addresses a field of bits, in this case the 32 bits which form single-precision real number. Slicewise format offers more floating point performance potential than fieldwise format, but not necessarily more I/O throughput performance.

CM Fortran codes can be slicewise or fieldwise and employ the CM Fortran Utility library to do I/O operations. Paris codes are fieldwise and employ the CM file system (CMFS) library, which supports some advanced features such as buffered and stream I/O. The CM Fortran Utility library does not support buffered or streaming I/O operations.

In all cases, the test programs open a file descriptor, write to the file, then read the file and close it. The data read is compared to the written data for equivalency. CM reads and writes to the DataVault conform to the shape of the CM geometry during the I/O operation. In all tests, the geometry of the CM is 1-dimensional NEWS of N processors. Each test creates its own files to ensure geometries of read and write files are equivalent to machine geometry. Unless noted otherwise, results of tests are achieved with version 6.1 software. All CM Fortran tests were conducted with CMSS version 6.1.

Timings are conducted using the CM timer software facilities. The CM timer routines record time elapsed and time busy. The former is the wall clock time to execute the code between the timer start and timer stop, while the latter is the time the CM itself is busy executing the instructions. Calculations for all timings and transfer rates in this paper use the elapsed time. Tests under 6.1 contain data points for 4,194,304 and 8,388,608 virtual processors (VPs), while tests under 6.0 have a maximum of 2,097,152 (VPs). The timings presented in this paper are averages of separate test runs. Tests were conducted at various levels of system usage, ranging from unloaded to heavily loaded.

4 PARIS Performance Tests, Graphs and Conclusions

This section covers the methodology of the PARIS performance tests, the resulting performance graphs and their interpretation. It is more extensive than the analogous CM Fortran section primarily because PARIS and the CMFS library offer a richer set of routines for using and investigating the CM I/O system.

4.1 PARIS Performance Tests

The PARIS tests have only one I/O format, fieldwise, but the CMFS I/O library contains routines for doing several different types of I/O. The basic test concerns synchronous I/O on the DataVault. The test program source lines which do the work are as follows:

```
#define CHAR 8

  /* Write to DataVault */
  CM_timer_clear(0);
  CM_timer_start(0);
  if ((i = CMFS_write_file_always(par_fd, temp, CHAR*num_bytes)) < 0)
    CMFS_perror("write error");
  CM_timer_stop(0);
```

The call to `CMFS_write_file_always` has each virtual processor write `num_bytes` bytes. The timers are wrapped around this call, so only the actual write is timed. The write is a synchronous I/O operation. The read call is handled exactly the same as the write, except for the function name, and a different file descriptor in the call.

The above routine is an example of synchronous I/O on the DataVault. The other types of I/O operations available on the DataVault are streaming I/O and buffered I/O. Streaming I/O is beneficial for applications which can read or write a large amount of data, in an uninterrupted stream. A streaming read attempts to keep the CMIOC's FIFO buffers full, until there is no more data available for reading or writing. Streaming I/O is disabled when running under timesharing. Streaming I/O is not tested in this paper, though performance characteristics of streaming I/O may be estimated from the behavior of synchronous I/O. For more information on streaming I/O, see [2].

Buffered I/O gives better results for repeated transfers of small amounts of data, which can be buffered and transferred in larger chunks. The programmer designates a memory field to use as the I/O buffer. The call to `CMFS_buffered_write_file_always` writes data to the memory field when space is available, and flushes the buffer to disk when necessary. Several tests are conducted with buffered I/O to determine its efficiency. The buffered I/O test loop is shown below.

```
#define CHAR 8

  /* Set the I/O buffer size */
  if (CMFS_setbuffer(par_fd, iobuf, buflen) < 0)
    CMFS_perror("Write can't setbuffer");

  /* Do buffered write to DataVault */
  CM_timer_clear(0);
  CM_timer_start(0);
  for (times = 0; times < buflen/(CHAR*num_bytes); times++)
    if (CMFS_buffered_write_file_always(par_fd, temp, CHAR*num_bytes) < 0)
      CMFS_perror("write error");
  if (CMFS_flush(par_fd) < 0)
    CMFS_perror("Write can't flush buffer");
  CM_timer_stop(0);
```

The Buffered I/O test repeatedly writes 4-bytes at a time to a buffer of 512-bytes, until the buffer is full. The `CMFS_flush` call guarantees that the buffer is fully written to disk before stopping the CM timer.

4.2 PARIS Performance Graphs

The many graphs relating PARIS I/O performance are grouped for easy comparison. The following sections describe the major groupings.

4.2.1 Comparison Between 8K, 16K, and 32K Processor Performance

The first set of graphs (figures 2 thru 9) is presented as a summary of overall performance. These graphs compare the throughput of the I/O system at the three possible machine sizes. These graphs are semi-log plots with an x-axis range of 1,000 to 10,000,000 VPs. Their y-axes range from 0 to 32 megabytes/sec, to cover the entire performance range theoretically possible with the tested hardware configuration. The left-most point on each line graphed represents a VP ratio of 1.

4.2.2 Buffered I/O Results

The next two graphs (figures 10 and 11) illustrate the performance of buffered I/O. The buffer size is 512 bytes, and the tests do 128 consecutive reads/writes of 4 bytes/VP to the buffer. The buffer is flushed when it is full or when a `CMFS_flush` is called. The flush in the test is redundant, since the buffer should be flushed automatically after 128 reads/writes. The maximum speed this test could achieve is bounded by the synchronous tests of 512-byte/VP reads/write, since the flush operation is exactly a synchronous 512-byte/VP read/write. These graphs are the same format as the previous 8K, 16K, and 32K comparison graphs.

4.2.3 Performance Ranges for Individual Tests

The graphs devoted to a single machine size (figures 12 thru 17) report the time required to do a specific operation. These graphs are limited to CMSS 6.1 results. For each machine size (8k, 16k, 32k), at least seven repetitions of the tests are run. The repetitions are run at various times so that the state of the system ranges from quiescent to heavily loaded. The mean, maximum and minimum times of the repetitions are plotted. This format illustrates the range of performance values which may be expected in applications doing similar I/O operations. Only the graphs of the running times for tests of 4-byte reads and writes are shown in the performance section of this paper. The equivalent 512-byte graphs are uninteresting because the mean, maximum

Physical Proccessors	*Available Ports*	*Procs/Ports*
8,192 (Seqs 0,1,2,3)	1	8,192
16,384 (Seqs 0-1,2-3)	1	16,384
32,768 (Seqs 0-3)	2	16,384

Table 1: Ratios of physical processors to CMIO bus ports for the attachable sizes of the CM-2.

and minimum times are indistinguishable, and the performance information is the same as that presented in figures 2 thru 5. The mean, min and max lines for the 512-byte cases coincide because the startup costs and latency are small compared to the time for the data transfer.

4.3 Conclusions – PARIS Tests

- Performance levels for 1, 2 and 4 sequencers.

  The most obvious performance trend in the graphs which compare 8K, 16K and 32K processor tests is that the 8K sizes consistently outperform the 16K and 32K sizes. The 512-byte read and write cases show a disparity of up to 100% between 1 sequencer and 2 or more sequencers. In these cases, there is a penalty for using more processors.

  The explanation for this trend is that each physical processor in a VP-set doing an I/O operation must communicate with the DataVault through the I/O ports connected to a single CMIO bus. When one sequencer is communicating, the ratio of physical processors to available CMIO bus ports is 8,192. When two sequencers (the sets 0-1 and 2-3 are the two possible sets of 16K processors) are communicating with the I/O bus, only one CMIO station id is available on a given CMIO bus (see figure 1). The ratio of physical processors to available CMIO bus ports is 16,384. For the 32k case, the possible set of sequencers is 0-3, and there are two station ids available on one CMIO bus (in figure 1, CMIO bus 1 is connected to sequencers 1 and 3, which are both in the set of 32K physical processors active). The ratio of physical processors to available CMIO bus ports is again 16,384. Table 1 summarizes the possible configurations. The ratios explain why the 8K results are up to twice as fast as the 16K results, and why the 16K and 32K results are nearly the same.

- Difference in performance between CMSS 6.0 and CMSS 6.1.

  The comparisons between the timings done under version 6.0 and version 6.1 suggest that optimizations made to improve throughput in version 6.1 were successful. The most dramatic improvement is seen in the 4 byte write test, which was the least efficient area under CMSS 6.0. Many of the 4-byte write data points indicate an improvement of 100% in throughput. The 4-byte reads improved by 5%–10% depending on the VP ratio and number of physical processors. The 512-byte reads and writes had less room for improvement, since at higher VP ratios, these types of operations were already very efficient under CMSS 6.0. For the most part, 512-byte read/write improved slightly, except for the 512 byte writes between 32K VP and 256K VP which are slightly slower (by less than 5%) under 6.1 than they were under 6.0.

  Overall, the improvements between CMSS 6.0 and CMSS 6.1 are significant, and the gains in the 4-byte writes offset the partial deterioration of the 512-byte write cases.

- Efficiency of Buffered I/O performance.

  The buffered I/O scheme is a useful alternative to writing small amounts of data at a time. The buffered I/O performance is very close to non-buffered I/O performance of the 512-byte tests, which was the size dedicated to the I/O buffer. Comparing the figures for the 512-byte unbuffered writes (figure 2), the 4-byte unbuffered writes (figure 6) and the

buffered writes (figure 10) shows that the buffered writes are 2 to 3 times faster than the unbuffered 4-byte writes. Similarly, the buffered reads are faster than the unbuffered 4-byte reads, and slightly slower than the unbuffered 512-byte reads.

- Differences between 4-byte and 512-byte reads and writes.

  The disparity between transfer rate of small (4-byte/VP) reads/writes and large (512-byte/VP) reads/writes can be quite large (up to 20MB/sec). The 4-byte read/write is an approximate measure of DataVault startup and overhead costs, since it is near the smallest amount of data which can be transferred. In the 4-byte cases, there are fluctuations between the mean, min and max, whereas the 512-byte cases do not have these fluctuations. The time to read/write the data for the 512-byte cases dominates factors such as disk head positioning and seek time, minimizing the presence of these costs in the graphs.

- Behavior of I/O at a wide-range of virtual processor ratios.

  Overall, the I/O operations perform in an acceptable range for 4-byte and 512-byte sizes, for all VP ratios. For a constant number of physical processors, the CM I/O system is most efficient when it does the read/write at a high VP ratio. Under CMSS 6.1, at a VP ratio greater than 2, increasing the VP ratio results in a higher throughput. Throughput above 20MB/sec for read and writes is available at VP ratios above 8 when using either buffered or synchronous 512-byte reads/writes.

- Consistency of performance between sequencers.

  The performance of sequencers on CMIO bus 1 is no different from the performance of sequencers on CMIO bus 2. To prove this, a series of tests are run on sequencer 2 and sequencer 1. The mean difference between times on sequencer 1 and sequencer 2 for a given VP ratio is less than 0.01 secs. These tests indicate that the performance of the two CMIO busses is essentially identical, and there is no tangible performance difference between equal size processor sets.

5 CM Fortran Performance Tests, Graphs and Conclusions

The first two parts of this section explain the CM Fortran I/O tests, and discuss the performance graphs. The third part presents conclusions on CM Fortran's I/O facilities.

5.1 CM Fortran Performance Tests

The I/O tests in Fortran are compiled for slicewise execution, since this is how CM Fortran codes achieve the best computational performance. The two types of storage format available under CM Fortran Release 1.1 are slicewise format and fieldwise format. The emphasis of the CM Fortran tests is to check that the DataVault can be effectively used from slicewise mode. As a comparison, however, tests are also run from slicewise Fortran using fieldwise I/O format.

Fieldwise uses the same storage format as the PARIS based I/O, and requires transposing to be compatible with slicewise Fortran executables. Therefore, fieldwise I/O is expected to be less efficient than slicewise I/O when compiled under slicewise mode. Table 2 shows the naming differences between slicewise (`_fms`) and fieldwise I/O operations. For more information on the CM Fortran Utility library I/O calls, see [1].

5.2 Graphs of Performance Results – CM Fortran Tests

The four graphs in this section cover CM Fortran I/O throughput under slicewise-compiled mode. The first two graphs, figures 18 and 19 show the range of performance using the CM Fortran Utility library FMS I/O calls. These calls write/read the data in the slicewise format.

Operation	*Slicewise*	*Fieldwise*
READ	`CMF_array_from_file_fms`	`CMF_array_from_file`
WRITE	`CMF_array_to_file_fms`	`CMF_array_to_file`
Open File	`CMF_file_open`	`CMF_file_open`

Table 2: Comparison of slicewise and fieldwise operations in the CM Fortran utility library.

The next two graphs, figures 20 and 21 show the performance for writing and reading using the fieldwise format I/O calls (see table 2).

Figure 18 shows that I/O rates above 10MB/sec are most easily obtainable when more than 10 megabytes are being written. Notice the steep increase in throughput between the 1 megabyte and 10 megabyte sizes, for all sequencer sets. This indicates that the total amount of data transferred is more influential than data per processor in determining throughput. The 8k performance line in figure 18 shows that writes of sizes less than 8MB resulted in transfer rates under 7 MB/sec. The 16k cases achieved better than 7MB/sec at all sizes above 1MB, with the peak rates near 20MB/sec for 8MB and above. The 32K cases were about 6MB/sec slower than the 16K sizes at all points above 128K.

The corresponding read performance, illustrated in figure 19 shows characteristics similar to the write performance, with the steep change shifted to the 100Kbyte and 1 Mbyte range. In the read statistics, however, all three sequencer sizes display relatively close levels of throughput at the various transfer sizes, whereas the write performance is more dispersed. At sizes above 128K the read transfer rates were above 15MB/sec for all three configurations: 8K, 16K and 32k. The 16K and 32K sizes peaked near 20MB/sec for reads larger than 8MB, while the 8k cases reached 25MB/sec for large reads. One explanation for the difference in performance at the top range on both the write and read graph is described in detail under the section: PARIS Performance levels for 1,2 and 4 sequencers.

The fieldwise graphs, figures 20 and 21, show that fieldwise performance is rather dismal under slicewise mode. When transferring over 128 Kbytes of data, the CMFS fieldwise I/O calls are at least 3 times slower than the comparable slicewise operations under slicewise CM Fortran. This is attributable to the transposing and data manipulation required to put slicewise data into fieldwise format. For a given size, the transposing to fieldwise format takes approximately twice as long as the I/O operation. For writes in fieldwise format, performance is never better than 4MB/sec for any size test. For reads, performance reaches nearly 7MB/sec for 1MB reads with 32K processors, but tails off for the next larger size.

5.3 Conclusions – CM Fortran Tests

There are several conclusions relevant to CM Fortran users which can be drawn from the CM Fortran I/O performance graphs. These conclusions and the differences and similarities between CM Fortran performance and PARIS performance are discussed in the following list.

- Slicewise users should use FMS calls.

  Comparing the slicewise write and read graphs (figs. 18 and 19) to the fieldwise graphs (figs. 20 and 21) shows that the FMS slicewise I/O calls are slightly more efficient for small arrays, and much faster on larger arrays (sizes greater than 1Mbyte). The fieldwise calls generally run out of memory for arrays larger than 10Mbytes which makes comparison above that level impossible. In any case, slicewise CM Fortran I/O users should use the FMS calls whenever compatibility with fieldwise format is not needed.

- CM Fortran I/O Performance is satisfactory for large arrays.

  For arrays larger than 1 million floating point elements, CM Fortran I/O performance is generally good. For write operations, one may expect 5-20 MB/sec throughput. For

read operations, 15-25 MB/sec throughput is obtainable on all different sequencer sets. Generally, it is not efficient to write/read arrays smaller than 1 million single-precision floating point elements in slicewise mode. Note that the CM Fortran Utility library does not provide for buffered I/O or streaming I/O, as is possible in PARIS. The CMFS I/O calls could be called from Fortran for this functionality, but the performance would be hampered by the fieldwise nature of these routines.

- Differences in I/O behavior between CM Fortran and PARIS

  CM Fortran never reaches the 30 MB/sec and above levels obtainable with large PARIS-based transfers. The difference in efficiency between PARIS and CM Fortran is most likely attributable to the format differences, and the fact that the DataVault and CMIO systems were originally designed for fieldwise data. One advantage CM Fortran has over PARIS is that the throughput rates remain relatively close for all sequencers at the higher transfer sizes i.e. CM Fortran is more consistent than PARIS for different sequencer sizes.

- Similarities in I/O behavior between CM Fortran and PARIS

  Several of the conclusions of the PARIS section above also apply to the CM Fortran I/O behavior of the CM. The PARIS tests are more extensive, and the behavior is examined more closely below. Some examples of behavior which is safe to extrapolate to CM Fortran I/O are: consistency of performance between sequencers; behavior of I/O at a wide-range of virtual processor (VP) ratios (array sizes); and the basic explanation of throughput rate levels for 1, 2 and 4 sequencers.

6 Summary

Performance of I/O on parallel supercomputers is a key component of overall performance and usability of supercomputer applications. The Connection Machine DataVault I/O system is designed for peak transfer rates of 32MB/sec. The major limitation of the current DataVault configuration is its inability to use more than one CMIO bus to communicate between the CM and the DataVault. This often means that problems running on 2 or more sequencers of the DataVault will have a hard time duplicating the throughput rates of problems which run on 1 sequencer because the ratio of processors to CMIO bus connections limits efficiency on larger machine sizes. The actual performance of basic DataVault operations is close to peak performance rates. CM Fortran provides a basic set of I/O routines which can achieve performance in the 10-20 MB/sec range for arrays larger than 1 million single-precision floating point elements. In PARIS, at high VP ratios, performance rates between 20-30MB/sec can be expected. Overall, the DataVault is an effective I/O system that facilitates application control of I/O operations from the Connection Machine.

References

[1] Thinking Machines Corp. *CM Fortran User's Guide* Version 1.0, Appendix A. Cambridge, Mass., February 1991.

[2] Thinking Machines Corp. *Connection Machine I/O System Programming Guide* Version 6.1, Chap 3. Cambridge, Mass., October 1991.

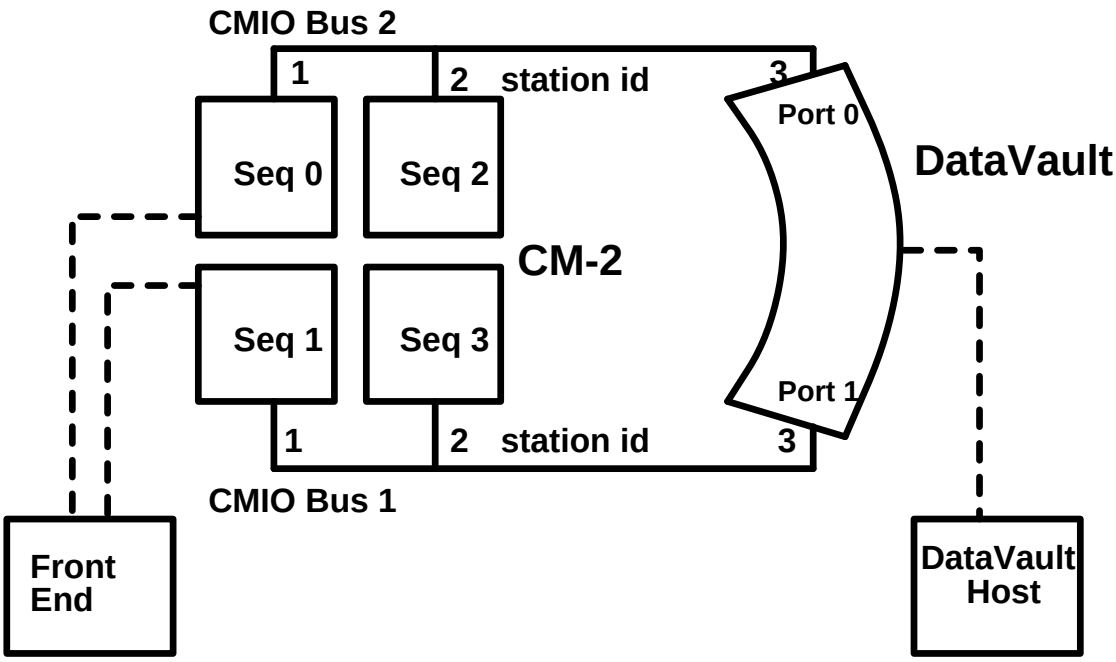


Figure 1: Basic architecture of CM I/O system. Two 32 MB/sec busses connect the sequencers of the CM to the ports of the DataVault. The two busses allow every sequencer to access the DataVault. Only one of the ports of the DataVault may be active at a time, so the maximum rate the DataVault may receive data is 32 MB/sec. Ethernet connections are represented with dashed lines. This figure is based on a configuration drawing by Mike Melendez of Thinking Machines Corp.

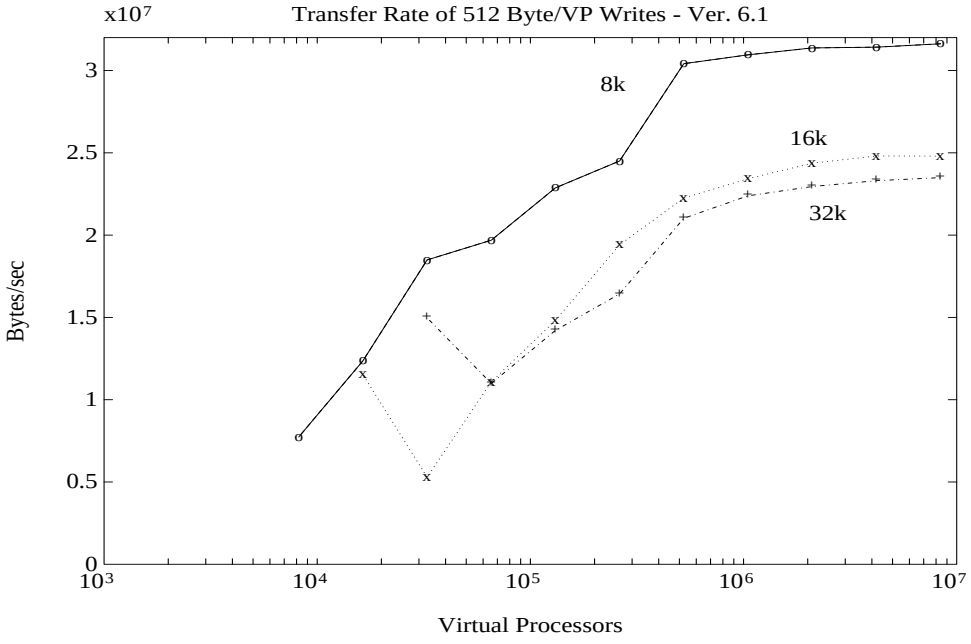


Figure 2: Version 6.1 PARIS transfer rates of writing 512-bytes per virtual processor for 8,192, 16,384 and 32,768 physical processors.

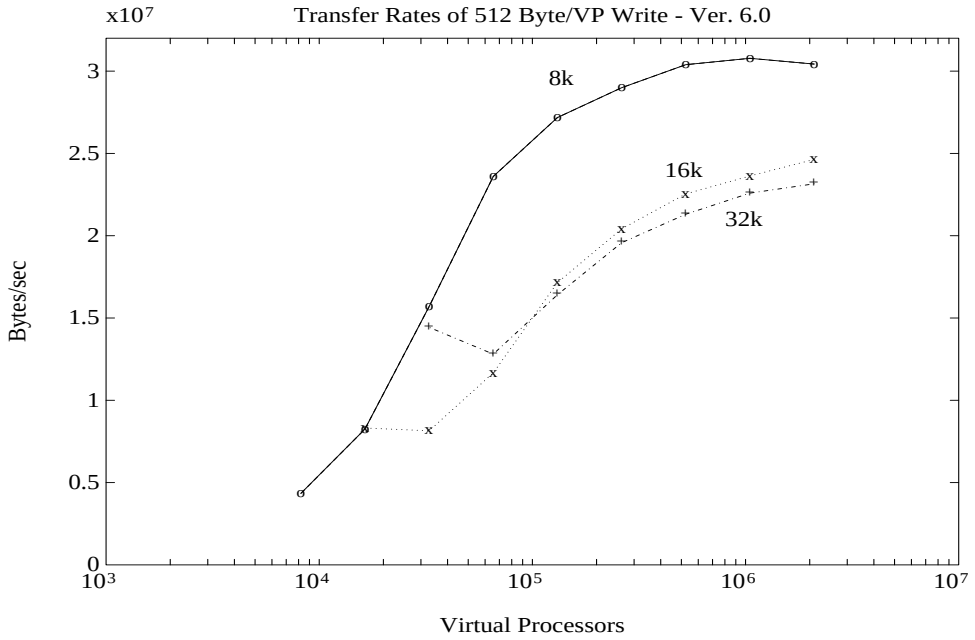


Figure 3: Version 6.0 PARIS transfer rates of writing 512-bytes per virtual processor for 8,192, 16,384 and 32,768 physical processors. Except for the 8K tests between 32,768 and 256K VPs, performance under 6.0 is slightly slower than under 6.1.

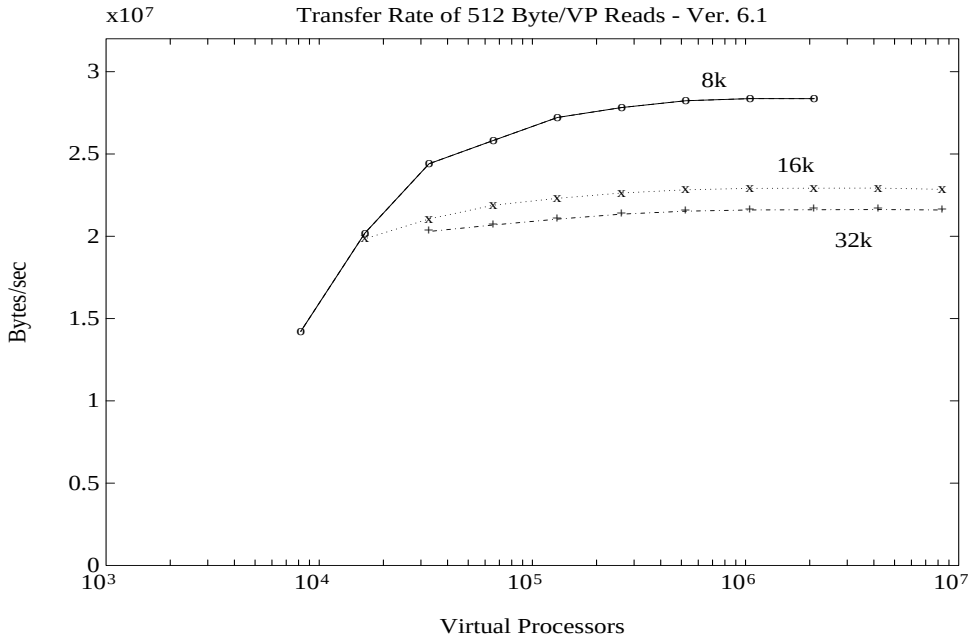


Figure 4: Version 6.1 PARIS transfer rates of reading 512-bytes per virtual processor for 8,192, 16,384 and 32,768 physical processors.

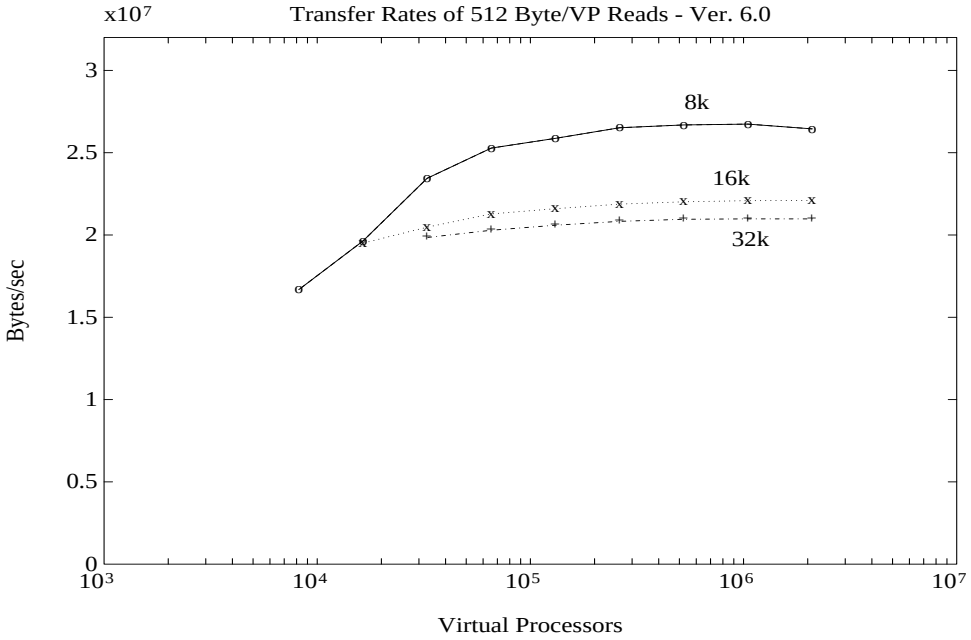


Figure 5: Version 6.0 PARIS transfer rates of reading 512-bytes per virtual processor for 8,192, 16,384 and 32,768 physical processors. Version 6.1 transfer rates are about 1-2 MB/sec faster than these in each case.

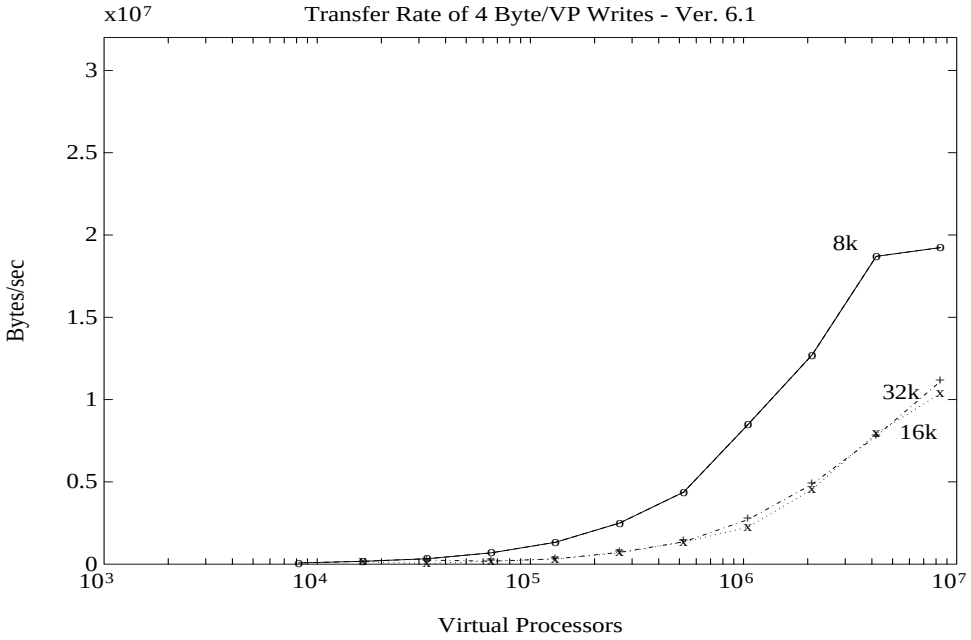


Figure 6: Version 6.1 PARIS transfer rates of writing 4-bytes per virtual processor for 8,192, 16,384 and 32,768 physical processors.

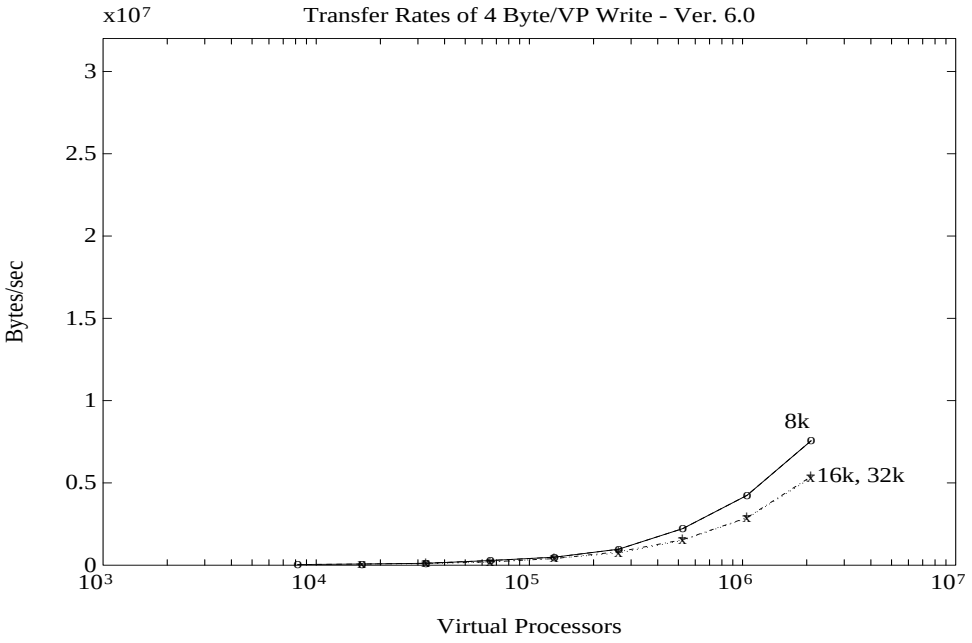


Figure 7: Version 6.0 PARIS transfer rates of writing 4-bytes per virtual processor for 8,192, 16,384 and 32,768 physical processors.

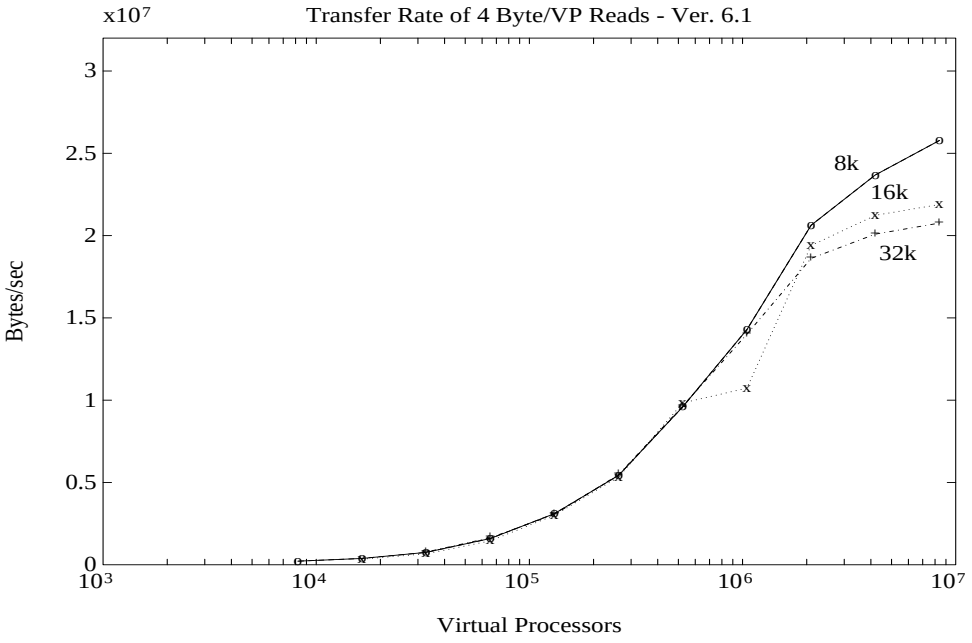


Figure 8: Version 6.1 PARIS transfer rates of reading 4-bytes per virtual processor for 8,192, 16,384 and 32,768 physical processors.

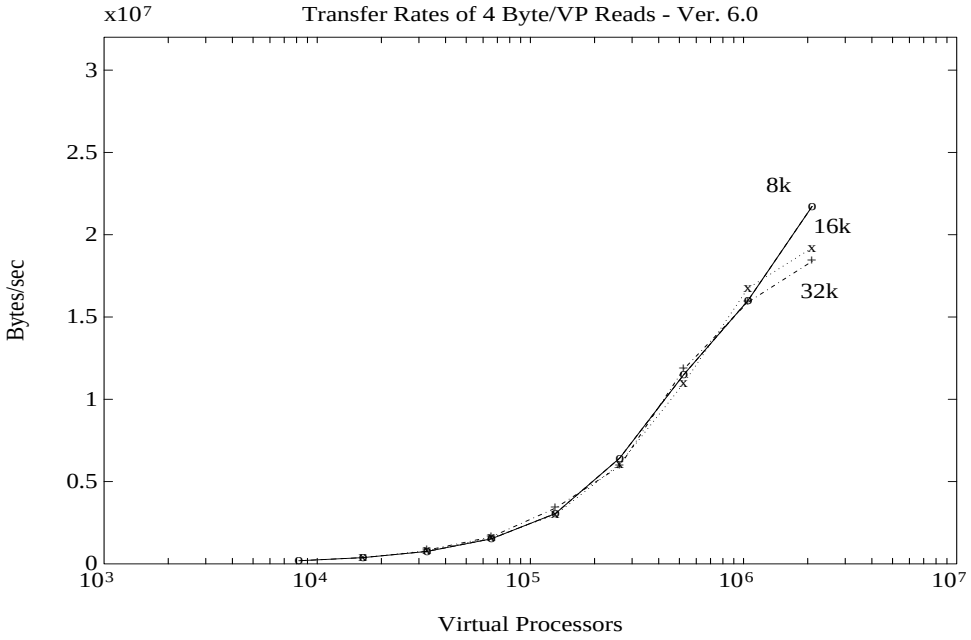


Figure 9: Version 6.0 PARIS transfer rates of reading 4-bytes per virtual processor for 8,192, 16,384 and 32,768 physical processors.

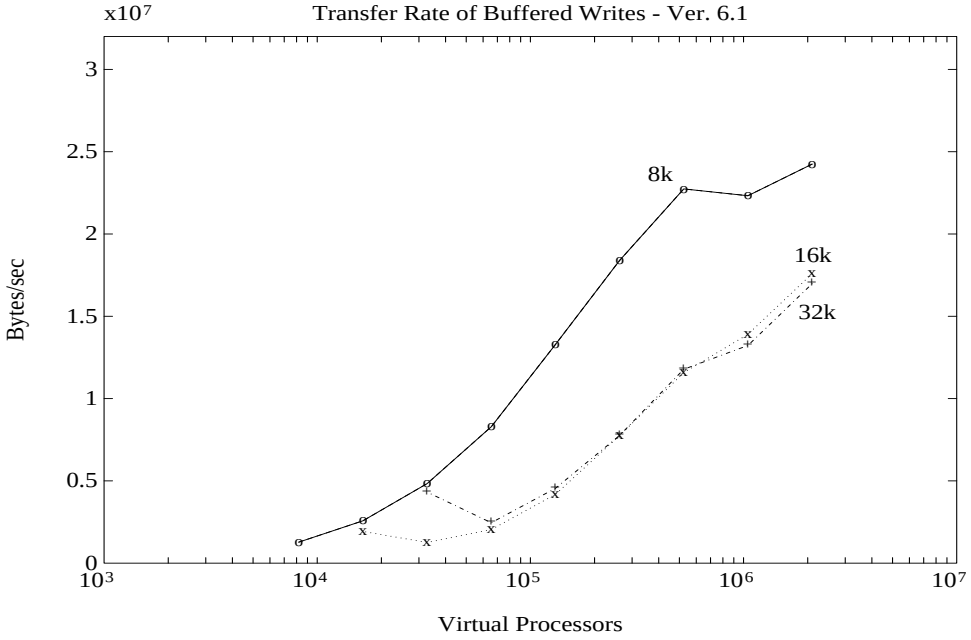


Figure 10: Buffered writes Version 6.1 (PARIS). Each buffer is 512-bytes.

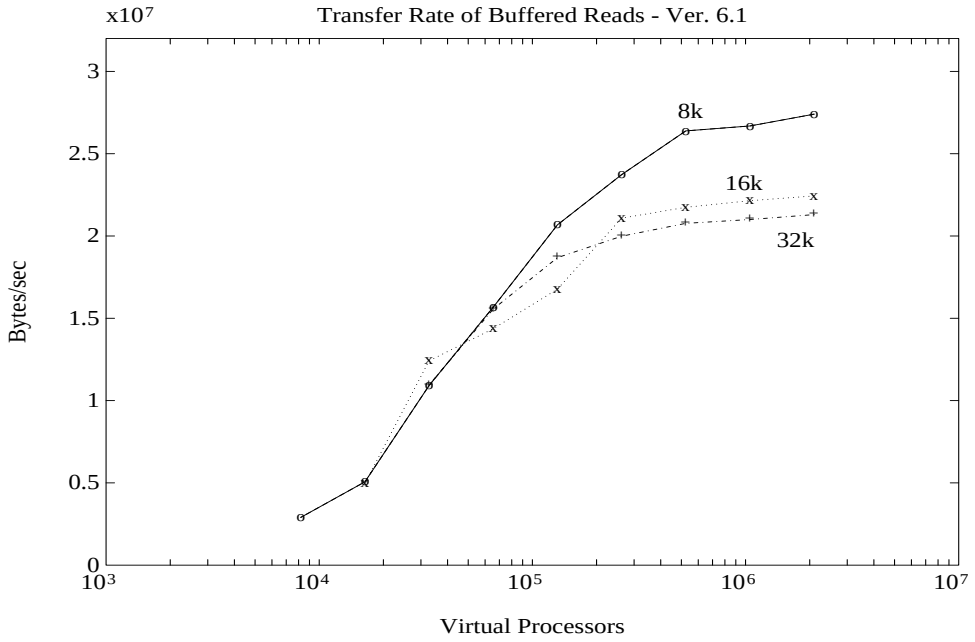


Figure 11: Buffered Reads Version 6.1 (PARIS). Buffer size is 512-bytes. Performance is close to that of 512-byte unbuffered reads under version 6.1 of CM system software.

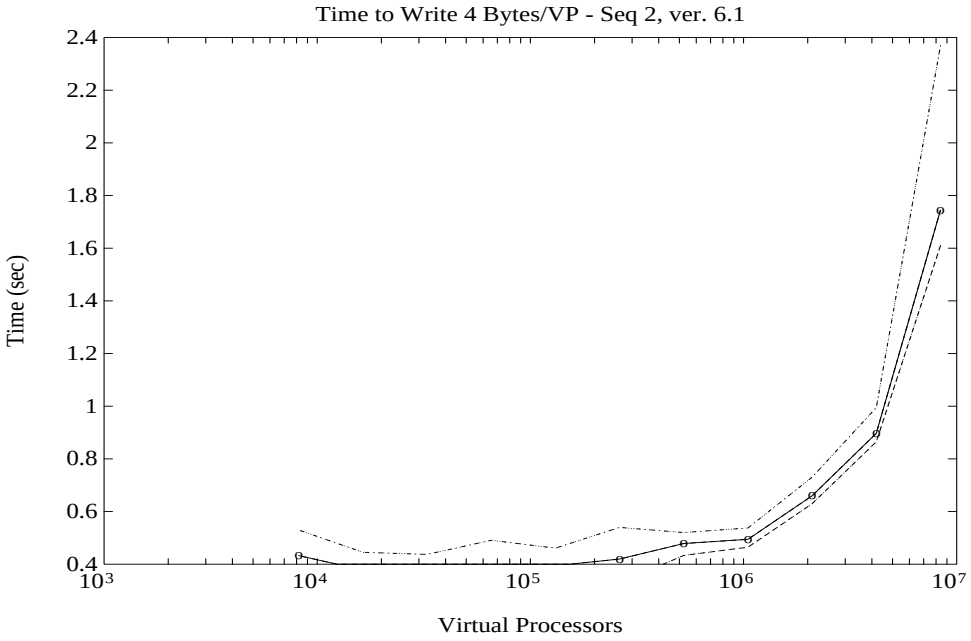


Figure 12: Transfer rate of 4-byte/VP writes on 1 sequencer, using CM system software version 6.1 PARIS. The solid line is mean time of 7 runs, with the upper and lower dashed lines being max and min times, respectively

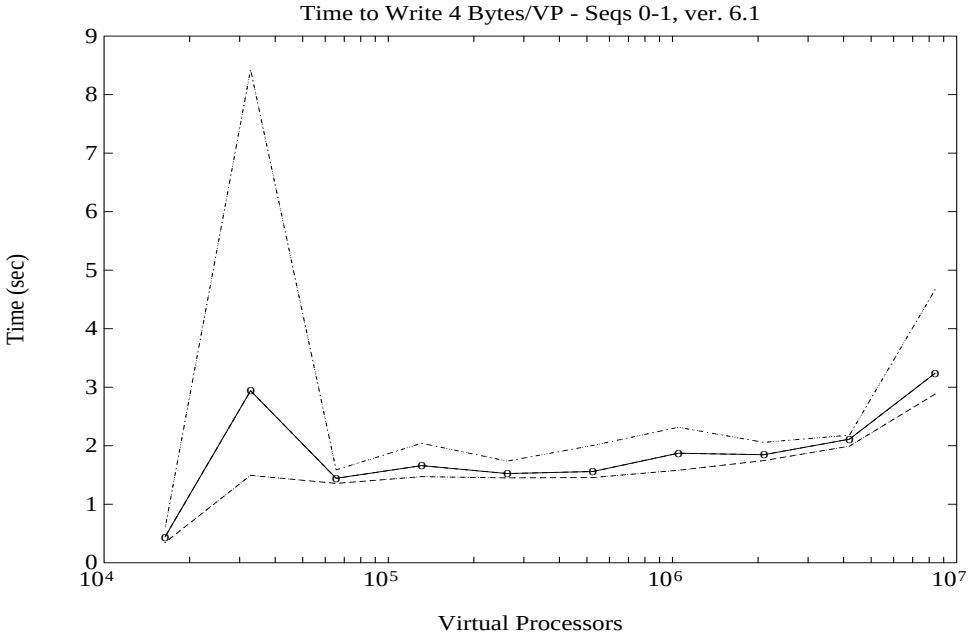


Figure 13: Transfer rate of 4-byte/VP writes on 2 sequencers, using CM system software version 6.1 PARIS. The solid line is mean time of 7 runs, with the upper and lower dashed lines being max and min times, respectively

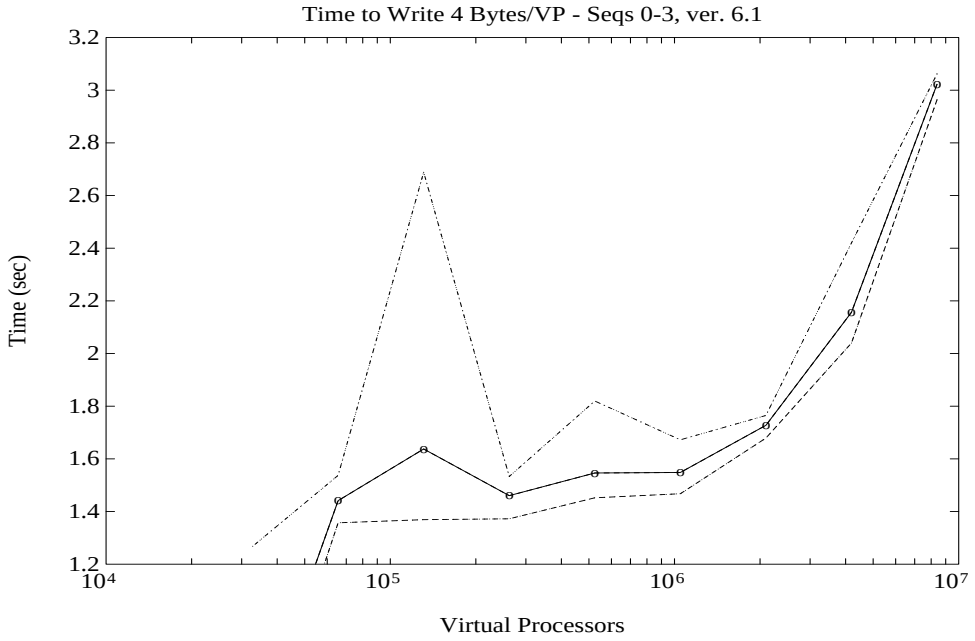


Figure 14: Transfer rate of 4-byte/VP writes on 4 sequencers, using CM system software version 6.1 PARIS. The solid line is mean time of 7 runs, with the upper and lower dashed lines being max and min times, respectively

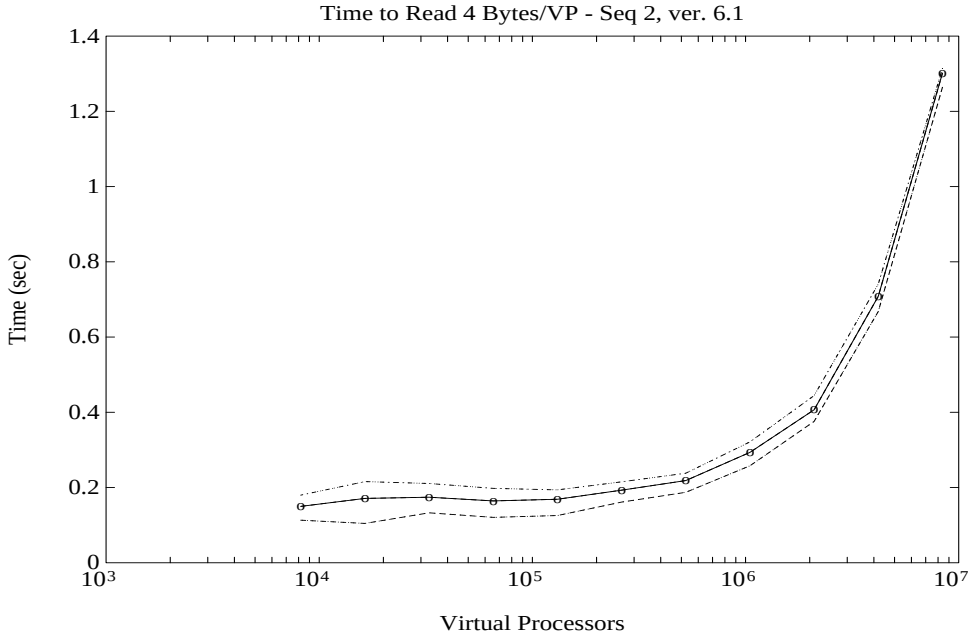


Figure 15: Transfer rate of 4-byte/VP reads on 1 sequencers, using CM system software version 6.1 PARIS. The solid line is mean time of 7 runs, with the upper and lower dashed lines being max and min times, respectively

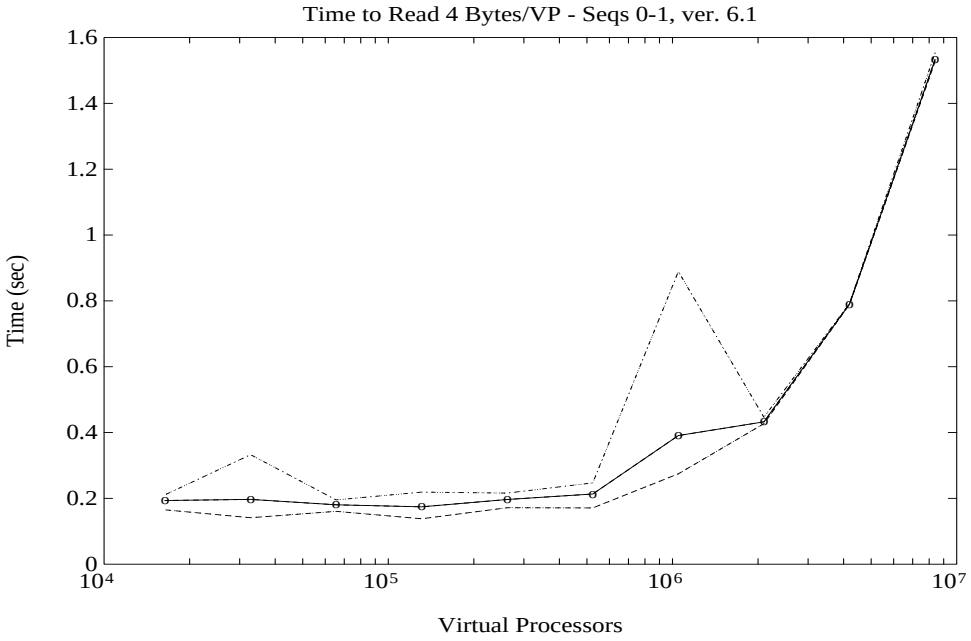


Figure 16: Transfer rate of 4-byte/VP reads on 2 sequencers, using CM system software version 6.1 PARIS. The solid line is mean time of 7 runs, with the upper and lower dashed lines being max and min times, respectively

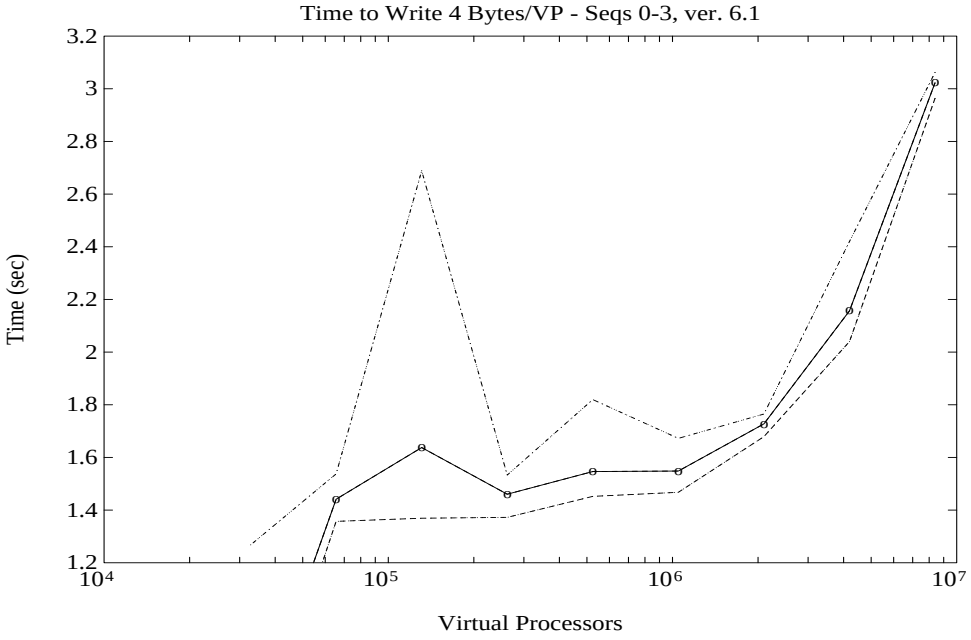


Figure 17: Transfer rate of 4-byte/VP reads on 4 sequencers, using CM system software version 6.1 PARIS. The solid line is mean time of 7 runs, with the upper and lower dashed lines being max and min times, respectively

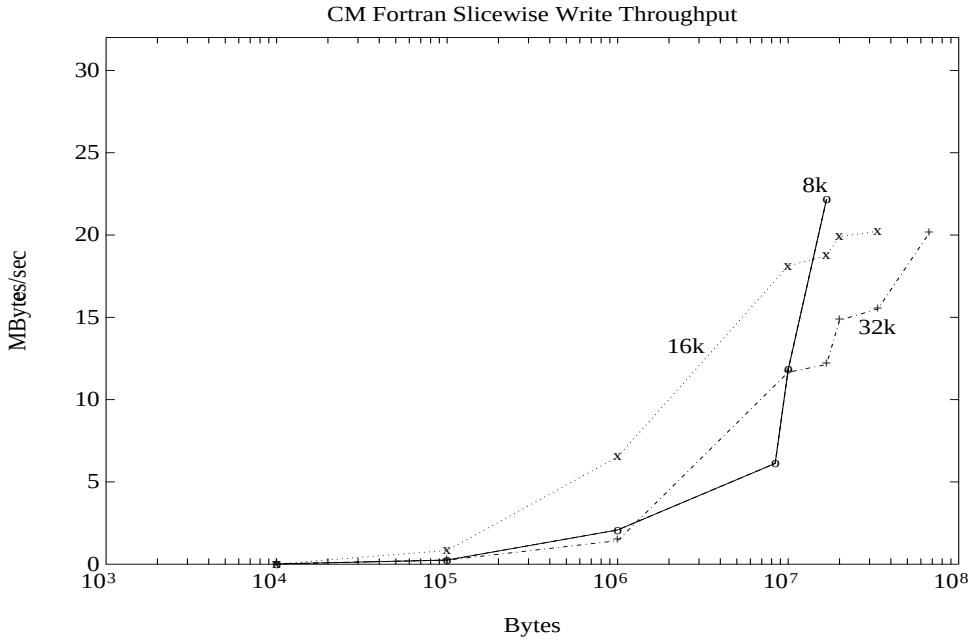


Figure 18: Transfer rates of writes, using slicewise FMS CM Fortran Utility library. Rates are shown for each of the three possible sequencer set sizes. The x-axis represents the size in bytes of the array written to the DataVault. Sizes range from 10Kbytes to 64Mbytes

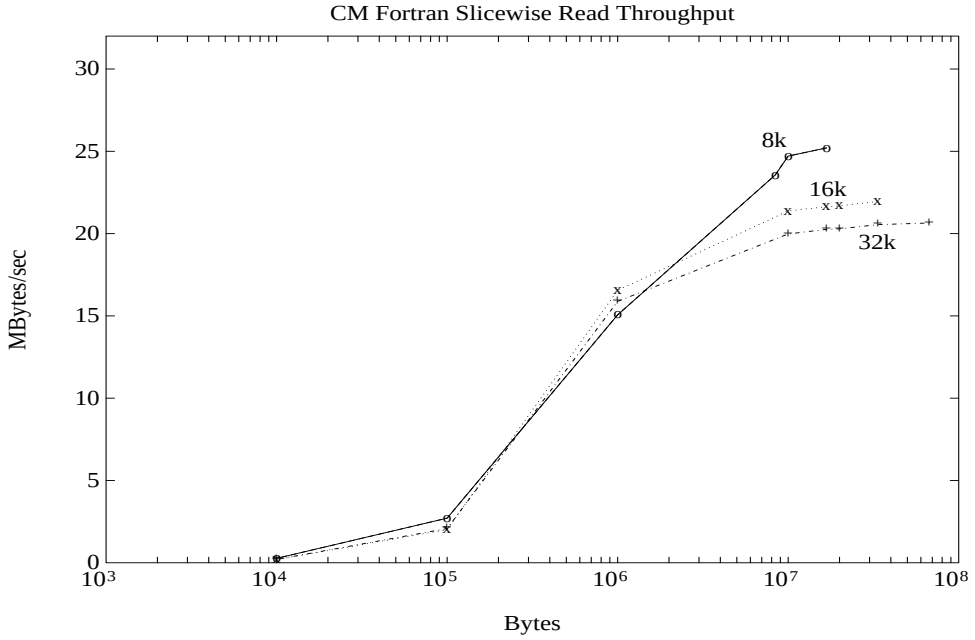


Figure 19: Transfer rates of reads, using slicewise FMS CM Fortran Utility library. Rates are shown for each of the three possible sequencer set sizes. Total read sizes range from 10Kbytes to 64Mbytes

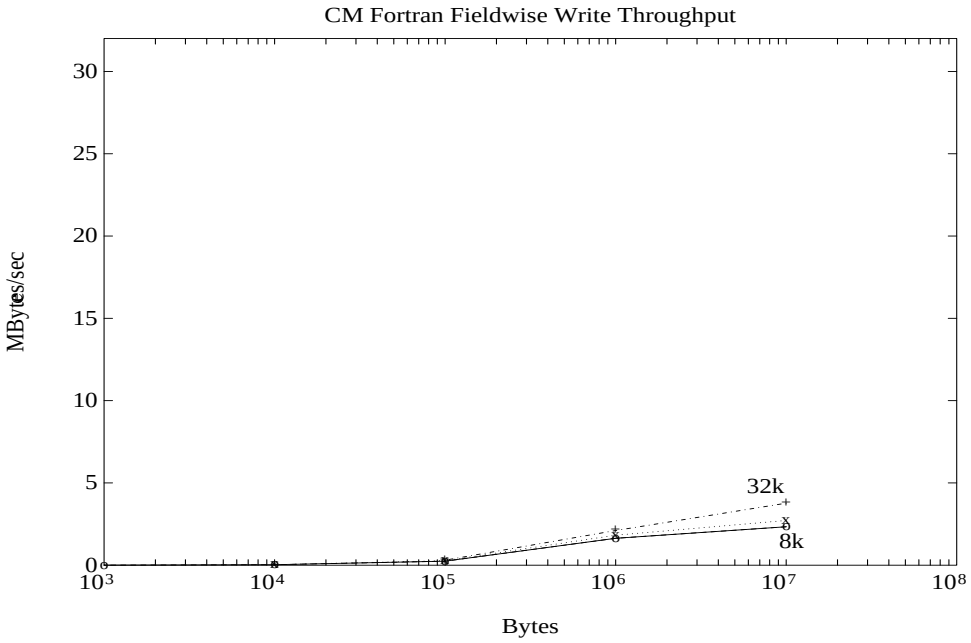


Figure 20: CM Fortran transfers rates of fieldwise writes, compiled in slicewise mode. Write sizes ranging from 1Kbytes to 10Mbytes are shown.

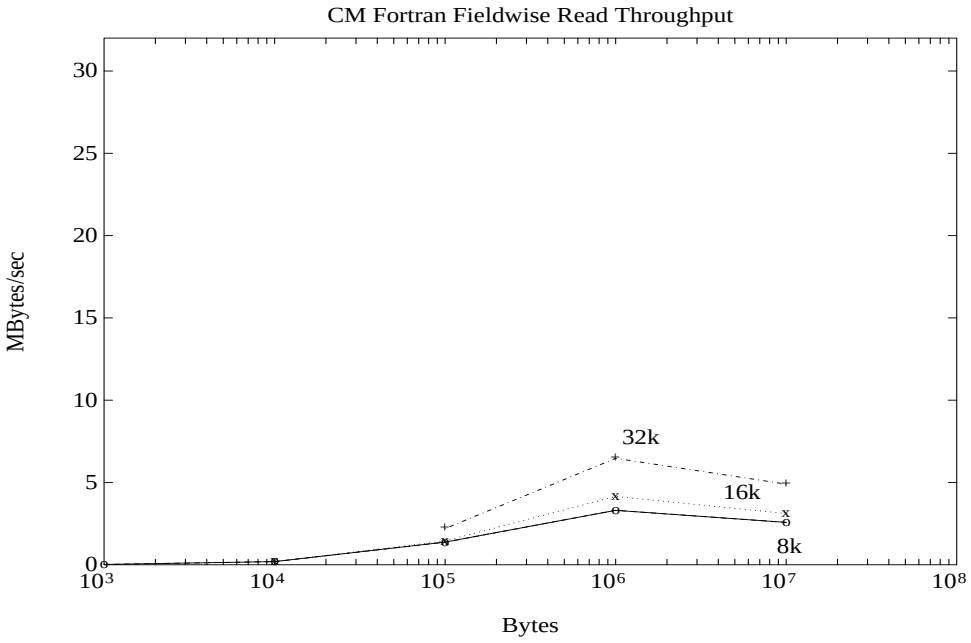


Figure 21: CM Fortran transfers rates of fieldwise reads, compiled in slicewise mode. Read sizes ranging from 1Kbytes to 10Mbytes are shown.