

Understanding the Performance of Molecular Dynamics and Quantum Mechanics Applications on Dell HPC Clusters

High-performance computing (HPC) clusters are proving to be suitable environments for running a wide range of parallel-processing applications. This article discusses the performance and scalability of two domain-specific scientific applications on a Dell™ PowerEdge™ HPC cluster running Linux: NAMD (for molecular dynamics) and DFT++ (for density-functional theory).

BY KALYANA CHADALAVADA AND SRIVATHSA NS

Over the last several years, the focus in high-performance computing (HPC) has shifted from massively parallel processors to symmetric multiprocessing (SMP) systems, and most recently, to loosely coupled, standards-based servers connected to form a cluster—popularly known as an HPC cluster. This trend has largely been driven by improvements in the performance of standards-based Intel® CPUs; advances in high-speed network interconnects; and efficient, standards-based message-passing libraries. The shift has also been accelerated by the rapid adoption of the Linux® operating system (OS) as a popular platform choice for clusters, and by improvements in compiler support for parallel-processing applications. This article explores performance and scalability factors of two popular scientific applications on Dell PowerEdge server-equipped HPC clusters: NAMD (for molecular dynamics) and DFT++ (for density-functional theory).

Factors that affect application performance on an HPC cluster

Several factors inherent to a cluster configuration can affect the performance of a parallel-processing application. Varying each factor can improve or degrade the performance of an application. This can make it a daunting task to arrive at the right combination of factors to provide optimal performance.

Factors that affect parallel-processing performance in HPC clusters include the following:

- **Compiler choice:** Compilers provide options to optimize the application code by applying several techniques, including techniques for exploiting the inherent parallelism in the code. Compilers can also generate object code targeted at a specific architecture. Using the right set of compiler flags while

generating the executable code can significantly affect the performance of the application.¹

- **Cluster topology:** The architecture of the cluster itself plays a major role in the performance of the application. The topology of the cluster and the interconnects used can affect inter-node communication.
- **System components:** The size of each component in the memory hierarchy (such as individual cache sizes and the main memory size), the presence or absence of Intel Hyper-Threading Technology (HT Technology),² the number of CPUs, the clock speed of each CPU, and the speed of the system bus all play a role in the performance of the application. HT Technology in particular can either enhance or deteriorate application performance depending on the nature of a particular application.
- **Middleware:** Most applications written for clusters are message-passing applications, which use the Message Passing Interface (MPI) standard³ for communication between individual processes. MPI implementations provide a handful of parameters that can be tuned to influence application performance.
- **Multithreading:** To exploit shared-memory parallelism, some applications either use explicit multithreading, allow the compiler to introduce threads, or use shared-memory programming paradigms such as OpenMP.⁴ How multithreading methods are used can affect the performance of an application.
- **Library choice:** For performance reasons, most applications use standards-based, third-party scientific and mathematical libraries. The way these libraries are built and configured can play an important role in performance.

NAMD application for molecular dynamics

Molecular dynamics simulations compute atomic trajectories by solving equations of motion numerically using empirical force fields—such as the Chemistry at HARvard Molecular Mechanics (CHARMM) force field—that approximate the actual atomic force in biopolymer systems. Simulation of large molecules requires enormous computing power.

NAMD (which stands for Not Another Molecular Dynamics program) is a parallel molecular dynamics code designed for high-performance simulation of large biomolecular systems.⁵ NAMD was designed to efficiently simulate large molecules on parallel-processing

systems. It is particularly well-suited to the increasingly popular Beowulf-class PC clusters—parallel-processing HPC clusters comprising industry-standard components. Based on Charm ++, a parallel object language based on the concept of virtual CPUs, NAMD is designed to scale to hundreds of CPUs on high-end parallel platforms and tens of CPUs on standards-based clusters.

DFT++ application for density-functional theory

A well-known and widely used generalized functional theory is density-functional theory (DFT), in which the energy of the system is parameterized as a function of electron density.

The DFT++ application provides code to perform complex density-functional calculations.⁶ In fact, DFT++ is a formalism for performing *ab initio* quantum mechanical calculations for density-functional-like systems. It is linear algebraic, basis independent, and fully explicit, making the transition from physical expression and mathematical derivation to computer coding easy and straightforward. The software is fully cache and register optimized, and runs in serial, threaded, MPI, and mixed-threaded MPI parallel environments. The software can perform both plane-wave pseudopotential (PWPP) and wavelet all-electron calculations within a unified framework, making it basis independent. It is also portable to a wide variety of environments.

Test bed configuration

In June 2004, Dell engineers performed tests on two HPC clusters—one comprising Dell PowerEdge 1750 servers and the other comprising Dell PowerEdge 2650 servers. To measure the effects of differing amounts of cache, the Dell team conducted tests on the PowerEdge 1750 cluster, then repeated the test runs on the PowerEdge 2650 cluster and compared those results with the results from the PowerEdge 1750 cluster. Other parameters, including processor speed and frontside bus (FSB) speed, remained the same. The clusters were configured as follows:

- Four Dell PowerEdge 1750 servers, each with dual Intel Xeon™ CPUs at 3.06 GHz with 1 MB of level 3 (L3) cache, 512 KB of level 2 (L2) cache, one 533 MHz FSB, and 1 GB of double data rate (DDR) RAM at 266 MHz
- Four Dell PowerEdge 2650 servers, each with dual Intel Xeon CPUs at 3.06 GHz with no L3 cache, 512 KB of L2 cache, one 533 MHz FSB, and 1 GB of DDR RAM at 266 MHz

¹For more information about compiler options, visit www.gelato.org/pdf/Compiler_options_final.pdf. Also, see “Execution Characteristics of SPEC CPU2000 Benchmarks: Intel C++ vs. Microsoft VC++” by S. T. Gurumani and A. Milenkovic in *Proceedings of the 42nd ACM Southeast Conference*, April 2004.

²For more information about Intel Hyper-Threading Technology, visit developer.intel.com/technology/hyperthread/index.htm.

³For more information about MPI, visit www-unix.mcs.anl.gov/mpi.

⁴For more information about OpenMP, visit www.openmp.org.

⁵For more information about NAMD, visit www.ks.uiuc.edu/Research/namd. Also, see “Biomolecular simulation on thousands of processors” by James C. Phillips, Gengbin Zheng, Sameer Kumar, and Laxmikant V. Kale, charm.cs.uiuc.edu/papers/NamdSC02.shtml.

⁶For more information about DFT++, visit dft.physics.cornell.edu. Also, see “New algebraic formulation of density functional calculation,” by Sohrab Ismail-Beigi and T.A. Arias, arXiv.org/abs/cond-mat/9909130.

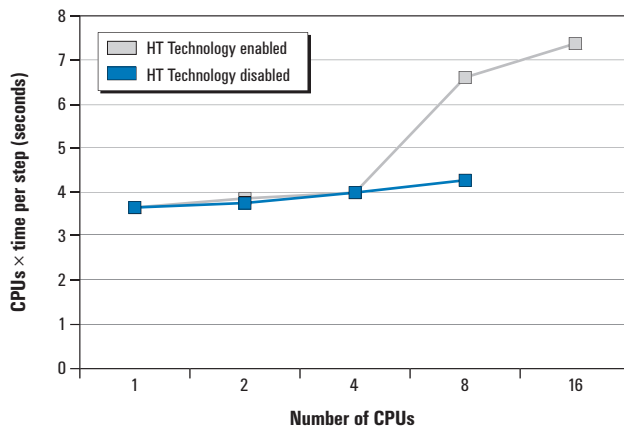


Figure 1. NAMD scalability

The two configurations—with and without 1 MB of L3 cache—were used to understand the dependency of the application on the memory subsystem. The test team used Gigabit⁷ Ethernet as the cluster interconnect. On-board Broadcom network interface cards (NICs) were used with a Dell PowerConnect™ 5224 switch. The team enabled and disabled HT Technology as necessary for the test runs.

The software was configured as follows:

- **OS:** Red Hat® Enterprise Linux 3, Update 2
- **Message Passing Interface:** MPICH version 1.2.5.2 from Argonne National Laboratory, configured with the P4 device and shared-memory communication
- **Compilers:** Version 8.0 Intel compilers with the following optimization options:


```
-O3 -xN -tpp7 -ipo
```
- **Libraries:** Fastest Fourier Transform in the West (FFTW⁸) version 2.1.5, compiled to support both double-precision and single-precision operations but without parallelization; Automatically Tuned Linear Algebra Software (ATLAS⁹) version 3.6.0 without support for threads
- **NAMD:** The Charm++ library compiled in an MPI implementation; input comprised the apolipoprotein A1 (APOA1) benchmark¹⁰ available on the NAMD Web site, used with default values
- **DFT++:** DFT++ version 3.0 with the profiling option turned on to obtain accurate timings for the runs; input comprised a sample data set with Planewaves used as the basis set

Test methodology

Dell engineers tested the runtimes of both NAMD and DFT++ on both clusters with the specifications discussed in the “Test bed configuration” section of this article. The team intended to discover how the applications would respond to specific configuration changes. The team then analyzed the results to understand the scalability of the applications, the effect of HT Technology on the applications, and the effect of cache on the applications. For both NAMD and DFT++, the team examined the computation times reported by the applications; the team did not measure the applications externally to determine the computation times.

Test results

The figures and the notations used in this article to describe test results for the two applications have subtle differences in format because of the nature of the applications; this article uses the graph style and format that best represents each application’s behavior. In addition, because DFT++ is a threaded application, performance curves are decidedly different from those of NAMD.

Scalability performance

For NAMD scalability, Dell engineers considered how the computation time for each step scaled with the number of nodes and the number of processes used for the computation on the PowerEdge 1750 cluster. The team observed results with HT Technology enabled and disabled. Figure 1 summarizes the results, plotting the number of CPUs against the product of the number of CPUs and time per step.

Whether HT Technology was enabled or disabled, the application scaled with the increase in CPUs. With HT Technology disabled, the performance curve was more horizontal, indicating that the application scaled particularly well with HT Technology disabled.

For DFT++, the test team observed the scalability of the application in nonthreaded mode using MPI across nodes with shared memory enabled and using several configurations of nodes and processes per node. Figure 2 shows how the application scaled.

As shown in Figure 2, a runtime improvement of nearly 73 percent occurred on a cluster of four nodes running two processes per node as compared to the baseline of one node running one process. Contrary to expectations, test results for two configurations—two nodes running two processes per node and four nodes running one process per node—were close to each other. This might indicate a low level of interprocess communication during the computation. Otherwise, the two-node configuration running two processes per node might have shown a significant performance gain compared to

⁷This term does not connote an actual operating speed of 1 Gbps. For high-speed transmission, connection to a Gigabit Ethernet server and network infrastructure is required.

⁸For more information about FFTW, visit www.fftw.org.

⁹For more information about ATLAS, visit math-atlas.sourceforge.net.

¹⁰For more information about the APOA1 benchmark, visit www.ks.uiuc.edu/Research/namd/performance.html.

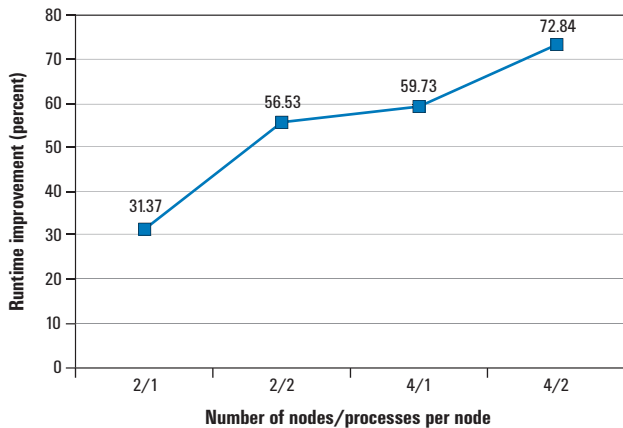


Figure 2. DFT++ scalability

the four-node configuration running one process per node because the two-process-per-node configuration used shared memory. The scaling was not close to linear, but showed a definite gain as the number of CPUs increased.

Effect of HT Technology on performance

The test team measured HT Technology performance using the PowerEdge 1750 cluster. For the NAMD application, the speedup attributed to HT Technology was calculated using the baseline of one node running one process to provide an indication of how HT Technology affected the performance of the application as the number of nodes/processes increased (see Figure 3).

For up to two nodes running two processes per node, the configuration with HT Technology enabled provided equivalent speedup to the configuration with HT Technology disabled. However, for four nodes running two processes per node, enabling HT Technology provided less speedup than having HT Technology disabled. When HT Technology was enabled, the speedup achieved with

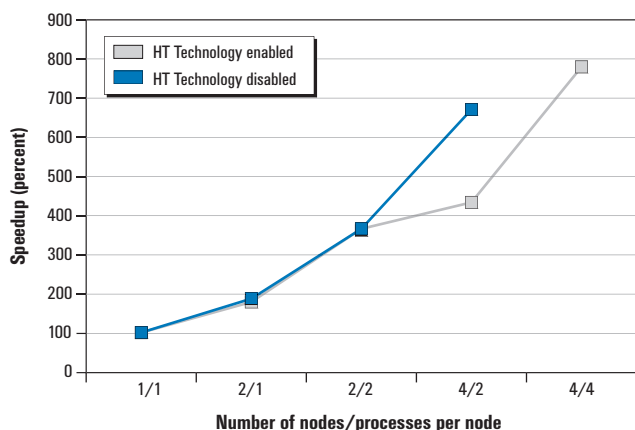


Figure 3. Speedup achieved for NAMD with HT Technology enabled and disabled

16 processes—four nodes running four processes per node—was better than the configuration of eight processes—four nodes running two processes per node—with HT Technology disabled. Similarly, four processes on two nodes (two processes per node) provided better performance than two processes on two nodes (one process per node). Because the speedup achieved with HT Technology enabled was greater than the speedup achieved with HT Technology disabled, the Dell team concluded that HT Technology can benefit NAMD when all the logical processors are used.

To observe the effects of HT Technology on DFT++, the test team ran DFT++ with threads enabled on a single node. Using a process with a single thread as the baseline, the test team varied the number of threads used for the computation, with HT Technology enabled and disabled in each case (see Figure 4).

When HT Technology was disabled, there were two physical CPUs and two logical processors. Increasing the number of threads from two to four did not lead to any performance improvements. This is indicated by the fact that the runtime when using four threads was nearly the same as the runtime when using two threads, with HT Technology disabled. When four threads were used, deterioration caused by scheduling contention did not occur.

When HT Technology was enabled, there were two physical CPUs and four logical processors, and the behavior changed dramatically with the number of threads. Using two threads, DFT++ performed better when HT Technology was disabled; the performance deteriorated when HT Technology was enabled. This deterioration in performance may be attributed to the fact that the threads were not bound to either the physical CPUs or the logical processors, thus leading to potential contention for resources if both threads were scheduled on the same logical processor. The Dell team observed the best DFT++ performance when running four threads with HT Technology enabled. In this case, each thread was run on a logical processor.

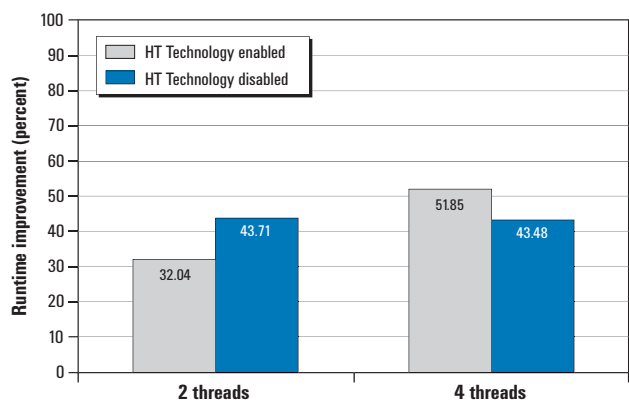


Figure 4. Effect of HT Technology on DFT++


In summary, HT Technology improved the performance of both NAMD and DFT++ applications in cases where all the logical processors were utilized. In cases where the number of threads was less than the number of logical processors, the behavior was unpredictable, sometimes leading to a deterioration in performance. This unpredictability can likely be attributed to the way that the OS scheduled the threads on the logical processors.

Effect of cache setting on performance

NAMD and DFT++ applications were executed on two similarly configured clusters: The first cluster used PowerEdge 1750 nodes and the second used PowerEdge 2650 nodes. The only other configuration difference between the clusters was that each node in the PowerEdge 1750 cluster had 1 MB of L3 cache, while the nodes in the PowerEdge 2650 cluster did not. The results obtained from the PowerEdge 2650 cluster were contrasted with the results obtained from the PowerEdge 1750 cluster to understand the dependency on the memory subsystem. For both applications, the performance was similar on both clusters. These results indicate that both NAMD and DFT++ have limited dependence on the memory subsystem.

Insight into factors affecting NAMD and DFT++ performance

By examining the specifics of NAMD and DFT++ performance described in this article, cluster architects may be better equipped to make performance-affecting decisions. The Dell test results reveal how both applications scaled when the number of CPUs varied. Additionally, testing on both applications led to valuable insights

about the effects of HT Technology settings; Dell engineers discovered that the way that threads or processes are scheduled on physical CPUs and logical processors can play a significant role in the overall performance of applications. 

Kalyana Chadalavada is a senior engineer with the Dell Enterprise Solutions Engineering Group at the Bangalore Development Center. Kalyana has a bachelor's degree in Computer Science and Engineering from Nagarjuna University in India. His current interests include performance characterizations on HPC clusters and processor architectures.

Srivathsa NS is a senior engineering analyst with the Dell Enterprise Solutions Engineering Group at the Bangalore Development Center. Srivathsa has a bachelor's degree in Science from Bangalore University, a master's degree in Computer Applications from Bharathidasan University, and a master's degree in Software Systems from Birla Institute of Technology and Science. During the past nine years, a significant portion of his contributions have been in the area of development environments for parallel computing, especially distributed debuggers.

FOR MORE INFORMATION

NAMD serial and parallel performance:

www.ks.uiuc.edu/Research/namd/tutorial/NCSA2002/pdf/14_performance.pdf

NAMD user's guide:

www.ks.uiuc.edu/Research/namd/current/ug

NAMD performance on a large IBM® cluster:

www.spscicomp.org/ScicomP9/Presentations/JHein_NAMD_on_Power4.ppt

Computational chemistry glossary:

www.shodor.org/succeed/compchem/glossary.html