Power Utilization vs. Application Performance on HP Servers Using Multi-core Processors—Conserving Application Energy

Dave Field, Dewayne McGehee, Don Mize, Chuck Newman, Sharon Shaw
Hewlett-Packard, 3000 Waterview Parkway, Richardson, TX 75080, USA

Introduction

There are many ways to optimize high performance computing workloads. In addition to the common approaches such as single job runtime, multi-job throughput, and parallel scalability, this paper discusses optimizing for power consumption. Measurements of power versus performance for standard benchmarks and ISV applications are provided.

Early in 2007, the HP High Performance Computing Division launched its Multi-core Optimization Program. The program's goal is to investigate and implement performance improvement techniques for HPC applications on HP servers that use multi-core processors. This power utilization analysis is a part of the HPCD program.

Terminology—HP uses "processor" to describe the physical chip and "cores" for the CPUs on each processor. We describe the server architecture with a notation combining the number of processors in the server and the total number of cores in the server. For example, 2p8c is a 2-processor server using quad-core processors with a total of 8 cores.

Configurations measured

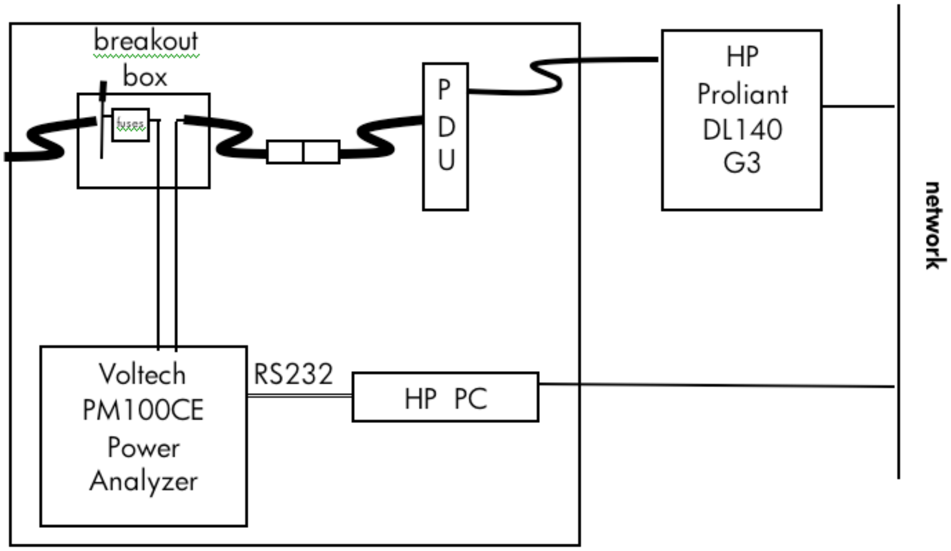
We chose Intel® Xeon® processors to demonstrate multi-core processor effects, since a dual-core Xeon processor and a quad-core Xeon processor exist with nearly identical functionality. We tested compute clusters of HP ProLiant DL140 G3 servers using these Intel Xeon processors. The specific configurations follow:

- DL140 G3, 2 dual-core Xeon 5160 3.0GHz processors, eight 1GB memory DIMMs—referred to as 2p4c (2 processors with a total of 4 cores), and
- DL140 G3, 2 quad-core Xeon 5355 2.66GHz processors, eight 2GB memory DIMMs—referred to as 2p8c.

The power measurement test bed

We wanted to correlate power measurements with application performance and to make it easy for our engineers to collect power measurements as they ran their codes. We developed the testbed shown in Figure 1. We selected the Voltech PM100CE Power Analyzer, since it met our capacity and accuracy requirements. To measure the power used by the HP server, we plugged the server into the power analyzer. The engineer added a script to the application initiation which signaled the power analyzer to begin measuring power and storing the measurements in a file on the PC. The sampling rate of the power adapter could be set, and after some experimenting, we decided that one sample per second provided a manageable amount of data and adequate granularity.

Figure 1. Power measurement testbed.



Some basic measurements

Before tackling ISV applications, we started with some basic measurements. We measured power utilization when the systems were idle. We measured power used running the LINPACK[1] benchmark and the STREAM[2] benchmark.

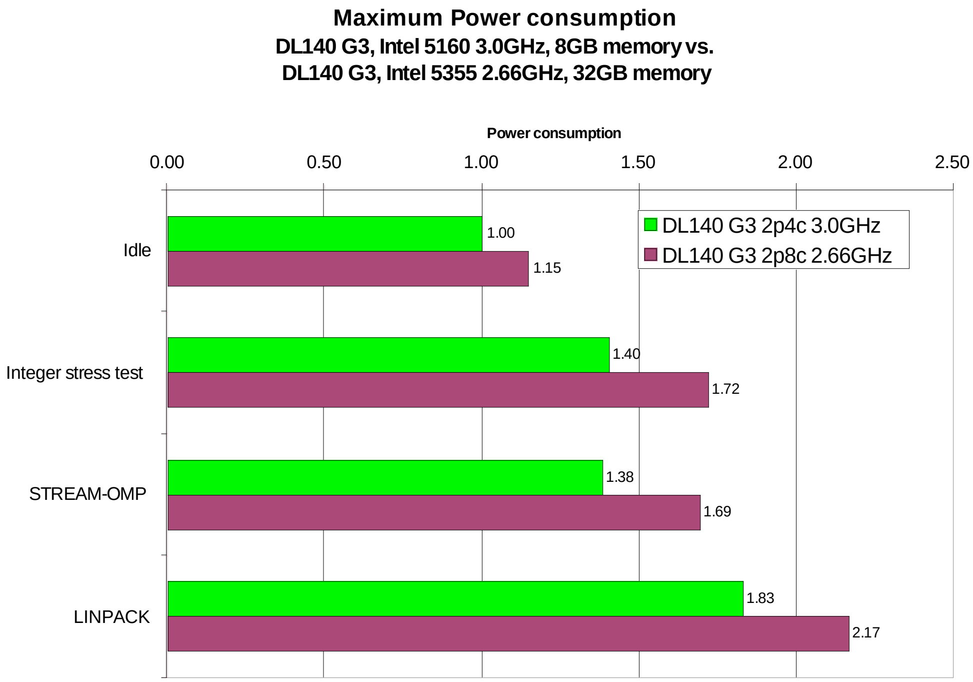
LINPACK is known to perform a very high rate of floating point operations. In an earlier power measurement project on HP Integrity servers, we demonstrated that LINPACK consumed more power than did any other code we tested. We wanted to know if this was also true for HP ProLiant servers using Xeon processors.

In fact, our tests on a range of ISV applications demonstrated that LINPACK consumes more power than any other code we tested. (LINPACK is a code that can be tuned for the specific configuration—in this paper, we used a tuned LINPACK, which has higher performance and higher power consumption than an un-tuned version).

STREAM performs a very high rate of memory operations and is often used to measure the memory bandwidth capacity of a server. It uses considerably less power than does LINPACK.

We also wanted to compare the power used by a floating point computational workload with that of an integer computational workload. We wrote a test code that performs a high rate of integer adds and multiplies. The power used by this integer math stress test was very close to the power used by STREAM.

Figure 2. Relative power utilization of basic measurements (relative to the idle 2p4c server)



The relative power between 2p4c and 2p8c configurations was similar for the following four tests:

	Ratio -- quad-core vs. dual-core
Idle	1.15
Integer stress test	1.22
STREAM-OMP	1.22
LINPACK	1.18

Sizing the power demand for your computer room

There are multiple sources of information about the power utilization of compute clusters. These estimates are based on maximum configurations and are likely to over-estimate the power requirements. If you run a typical HPC workload composed of floating-point-intense ISV applications, the LINPACK application can be used to size the power requirements of your compute servers.

Next, you need to know the power consumed by storage products. Power consumed by disks can vary considerably among products. In one test, a server-attached array of JBOD disks consumed 3.5 times more power than did the same number of disks internal to a server. The disks in the JBOD and the internal disks were SCSI U320 disks of the same capacity with approximately the same IO rates. The power requirement of your fileservers and storage is an important component of your facility's total power requirement.

Measurement variation

Power utilization is not an exact measurement. We ran LINPACK on several identical ProLiant DL140 G3 servers and found a variation of up to 3% among these systems. Always add a few percent to your measurements when sizing your power requirements.

HPC ISV Applications

We tested a cross-section of ISV applications:

- Abaqus Explicit from SIMULIA
- Abaqus Standard from SIMULIA
- ANSYS Multiphysics from ANSYS, Inc.
- FLUENT from ANSYS, Inc.
- MSC NASTRAN from MSC.Software
- Powerflow from Exa Corp.

Conserving Application Energy

Power utilization measurements are necessary to size the power requirements for a computer facility, but these measurements do not tell system managers enough to optimally use the power. It is also necessary to know the time duration over which the measured power will be consumed to run a given application.

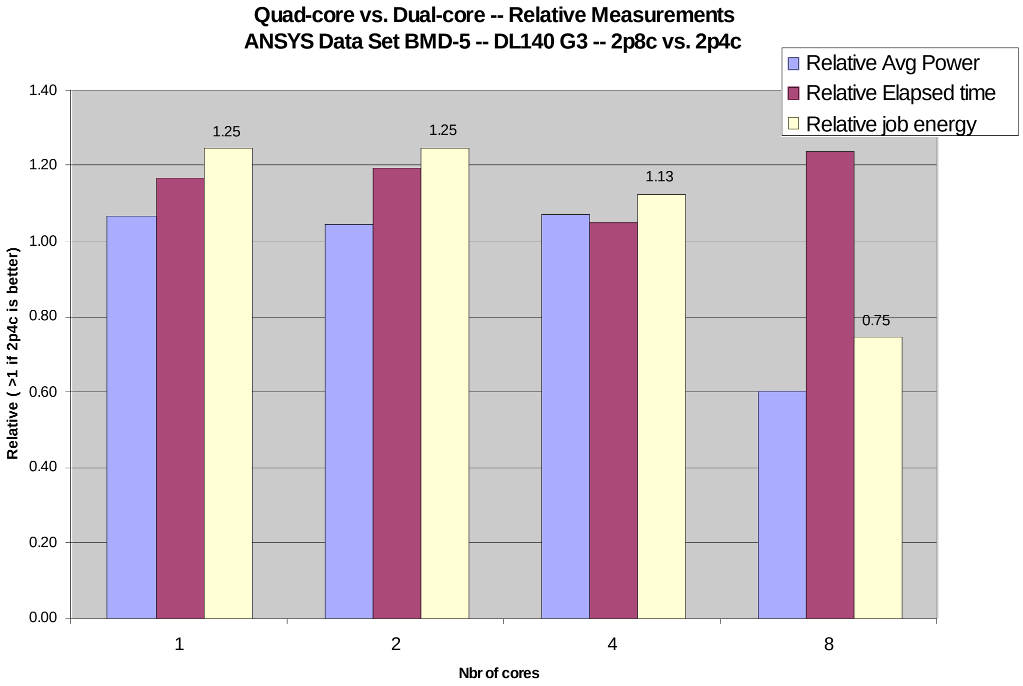
The product of the elapsed time of an application and its average power utilization is the Application Energy, which can be minimized, given enough data about the workload. Optimizing to minimize application energy is different than optimizing for fastest single job runtime or maximum throughput workload. However, application energy can be used to select the optimum processor type and system configuration; once determined, application energy can be used to optimize the work flow.

ANSYS[4] can be used to demonstrate two ways to use application energy: to differentiate a dual-core-based server from a quad-core-based server and to optimize the workload on the server. To create Figure 3, we ran one of the ANSYS performance benchmarks from 1-way (serial) to 8-way-parallel on the 2p4c server and the 2p8c server. We measured the runtime of the job and the average power utilization and computed the application energy. Then, for each level of parallelism, we divided the quad-core data by the dual-core data. In the figure, if a bar is > 1, then the 2p4c server outperformed the 2p8c server. ("Outperform" means using less power, running in less time, and using less application energy).

At 8-way-parallel, two 2p4c servers are required, whereas only one 2p8c server is required—as a result, the power ratio shows a substantial benefit to the 2p8c server.

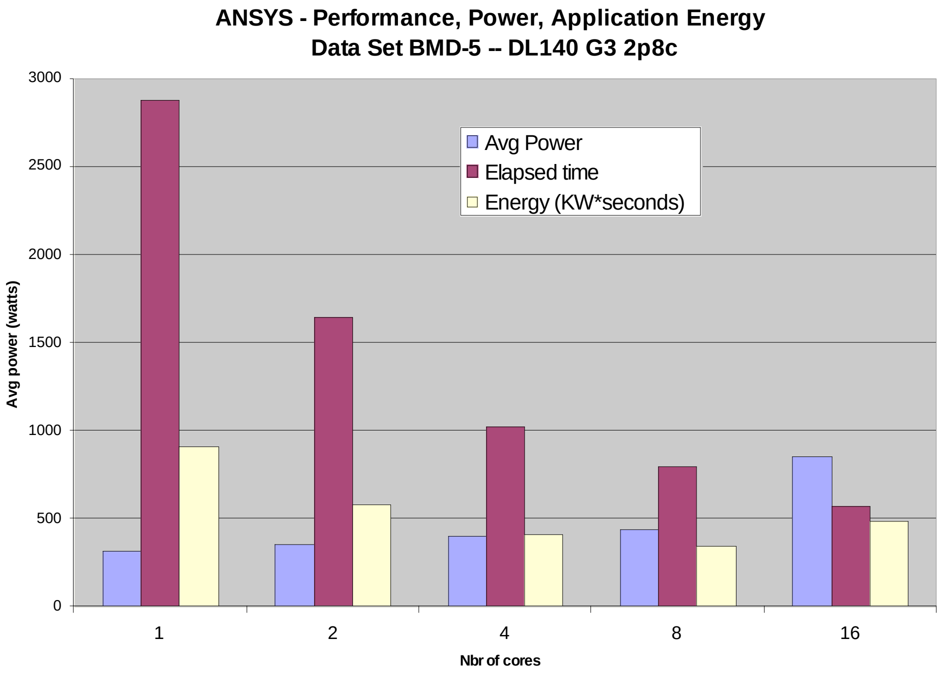
This data can be used to assist in decisions. For example, if the user wants to run this workload 8-way-parallel, the choice of server based on job runtime is easy—the 2p4c server is 1.22 times faster than the 2p8c server. But if you are optimizing for power consumption, notice that the 2p8c server consumed only 0.75X the power of the 2p4c server to complete the same job.

Figure 3. Comparing power, job runtime, and application energy on dual-core-based server vs. quad-core-based server.



Given a configuration, it is then possible to optimize the workload. For example, we want to decide the optimal level of parallelism for a job. Using the ANSYS example again, in Figure 4 we show the average power, job runtime, and application energy for the 2p8c server. If optimizing for runtime, 2-way or 4-way-parallel are excellent choices. Up to 4-way-parallel, speed ups in runtime are achieved. Beyond 4-way, the runtime gain per core is poor. But if you are optimizing for power consumption, notice that the job consumes minimum application energy when run 8-way-parallel.

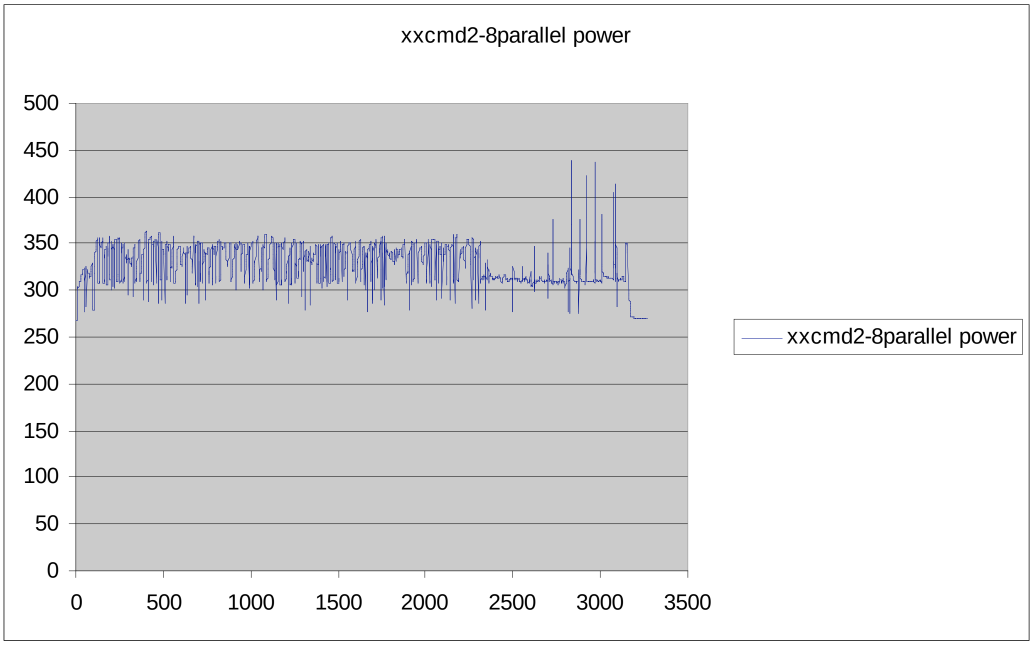
Figure 4. Comparing power, job runtime, and application energy on a 2p8c server.



Average vs peak power

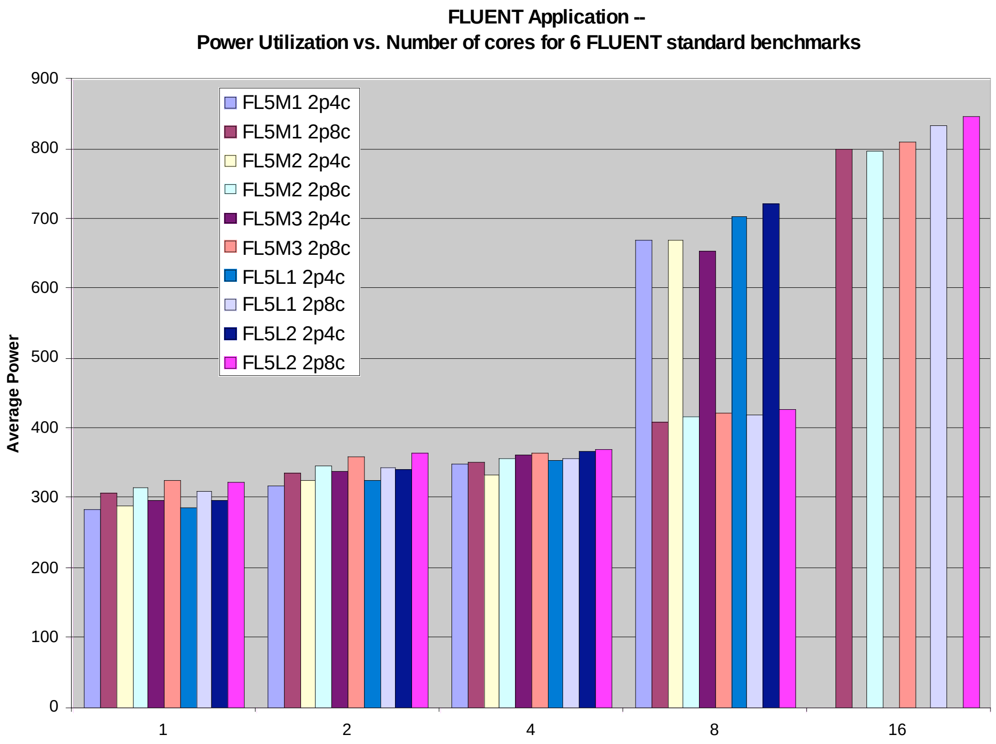
For each of the codes, we measured the average and maximum power used during the job. Both of these measurements are important because they provide both the steady-state and the peak power demands of codes. Many codes had tiny variations from average during their run–the maximum power was within 5% of the average power. A few codes that perform a large amount of filesystem IO, such as NASTRAN[5] and ANSYS, show large power variations. The peak power usage was up to 35% above the average.

Figure 5. Power variation during 8-way-parallel job using an MSC NASTRAN standard benchmark.



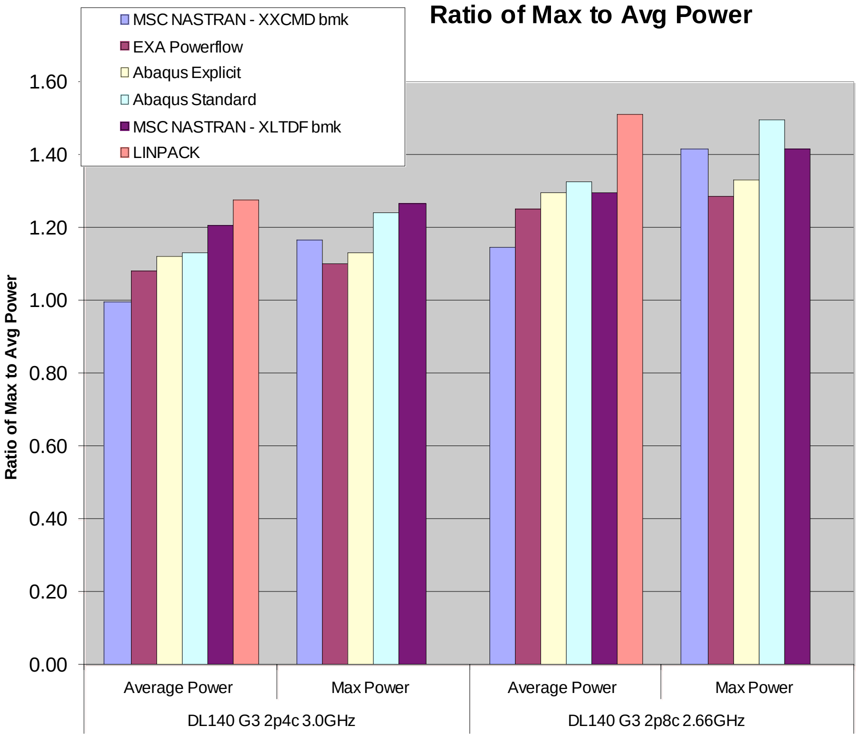
In addition to analyzing power variations during a single job, we also looked at power used by different input data sets for a given ISV application. As shown in Figure 6, some applications use approximately the same amount of power, independent of the input data set, making it possible to predict the power usage of any data set. In this example, we ran six standard benchmarks for the FLUENT[3] application on servers using dual-core processors (2p4c) and quad-core processors (2p8c), running from serial up to 16-way-parallel.

Figure 6. Power utilization of FLUENT application vs. standard benchmarks.



As shown in Figure 7, some applications vary little in their power usage during a single job and others vary considerably. To determine steady-state power requirements, average power usage is more important than maximum usage. However, it can be important to know the peak utilization. As this figure shows, no application uses more power than LINPACK–even the maximum power usage of ISV applications is less than the steady-state usage of LINPACK.

Figure 7. Range of power utilization in applications–max power vs. average power.



Conclusions

If minimizing application energy is a major goal for a computer facility, it is necessary to look at your workload in a different way. By measuring the performance and power utilization of your workload, you can identify energy-efficient ways to perform your work.

For any workload and cluster configuration, it is possible to determine a good estimate of power usage. Power utilization can be used in two ways–to assist in configuration trade-off decisions and to optimize the workload on a specific configuration.

Figures 8 and 9 show the average power used to run a set of codes on clusters of quad-core versus dual-core processors and also the ratio of power utilization on these two configurations. The average power utilization for all of these codes is 1.17 times more for the quad-core-processor-based cluster than for the dual-core-processor-based cluster. So, if the average job runtime is at least 1.17X faster per processor on quad-core processor versus dual-core processor, the quad-core-processor-based cluster is superior in application energy and in electricity cost.

Figure 8. Summary of power utilization of standard benchmarks and ISV applications.

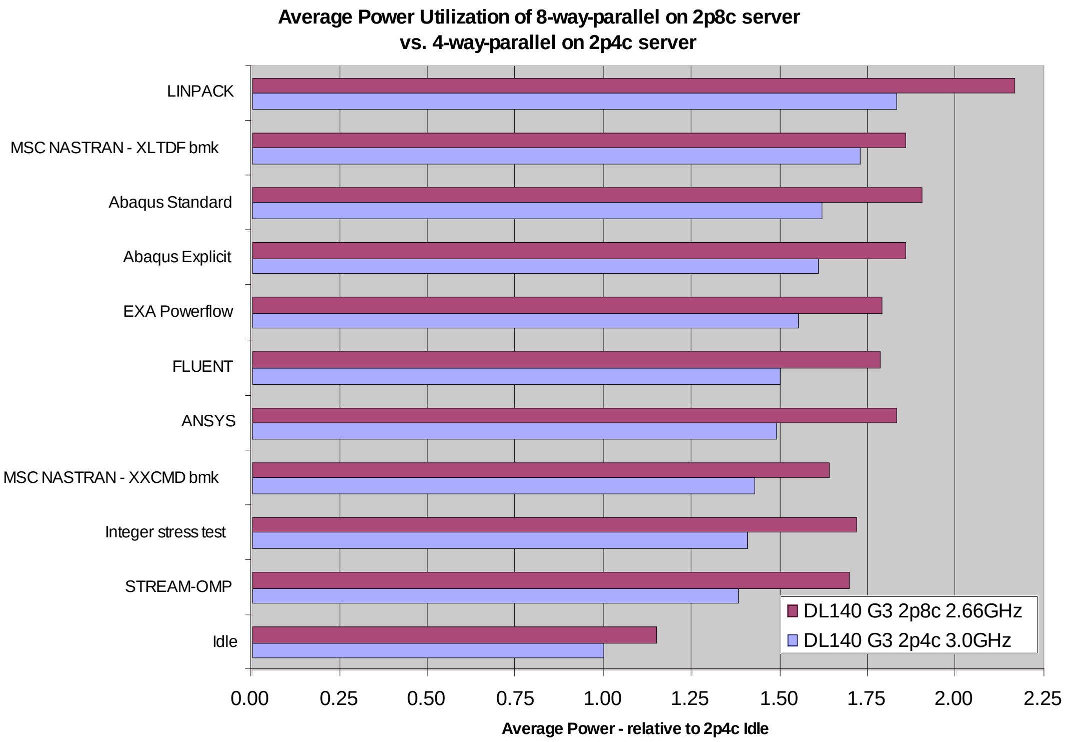


Figure 9. Ratio of average power utilization of 8-way-parallel on 2p8c server vs. 4-way-parallel on 2p4c server.

	2p8c vs 2p4c Avg Power Ratio
Idle	1.15
STREAM-OMP	1.22
Integer stress test	1.22
MSC NASTRAN - XXCMD bmk	1.15
ANSYS	1.23
FLUENT	1.19
EXA Powerflow	1.16
Abaqus Explicit	1.15
Abaqus Standard	1.18
MSC NASTRAN - XLTDF bmk	1.07
LINPACK	1.18

To obtain the optimum HPC cluster and optimize your workload, you need to decide on the optimization criteria. If power utilization is an important factor, then the above examples can assist you in guiding your technical evaluation.

Regardless of your criteria, the more you understand the performance characteristics of your workload, the better your decisions will be. If you look at the data in different ways, it can provide new insights about performance. In addition to performance per core, consider performance per processor, price/performance of the server cluster and the licensed applications, and application energy.

References

[1] “Performance of Various Computers Using Standard Linear Equations Software”, Jack Dongarra, University of Tennessee, Knoxville TN, 37996, Computer Science Technical Report Number CS - 89—85, url:http://www.netlib.org/benchmark/performance.ps.

[2] McCalpin, John D., 1995: "Memory Bandwidth and Machine Balance in Current High Performance Computers", IEEE Computer Society Technical Committee on Computer Architecture (TCCA) Newsletter, December 1995.

Acknowledgments

The idea for this project originated in HP’s High Performance Computing Division. It is one of the results of HP’s Multi-Core Optimization Program, which seeks ways to improve total application performance and per-core application performance on servers using multi-core processors.

For more information

[3] ANSYS FLUENT Performance benchmarks
http://www.fluent.com/software/fluent/fl5bench/

[4] ANSYS Multiphysics Performance benchmarks:
http://www.ansys.com/services/hardware-support-db.htm
and select Benchmarks

[5] MSC NASTRAN performance benchmarks:
http://www.mscsoftware.com/support/prod_support/nastran/performance
and select V2006 serial

www.exa.com

www.hp.com/go/hpc

4AA1-6090ENW, November 2007