ENERGY PROFILING AND ANALYSIS OF THE HPC CHALLENGE BENCHMARKS

Shuaiwen Song[1]
Rong Ge[2]
Xizhou Feng[3]
Kirk W. Cameron[4]

Abstract

Future high performance systems must use energy efficiently to achieve petaFLOPS computational speeds and beyond. To address this challenge, we must first understand the power and energy characteristics of high performance computing applications. In this paper, we use a power-performance profiling framework called PowerPack to study the power and energy profiles of the HPC Challenge benchmarks. We present detailed experimental results along with in-depth analysis of how each benchmark's workload characteristics affect power consumption and energy efficiency. This paper summarizes various findings using the HPC Challenge benchmarks, including but not limited to: 1) identifying application power profiles by function and component in a high performance cluster; 2) correlating applications' memory access patterns to power consumption for these benchmarks; and 3) exploring how energy consumption scales with system size and workload.

Key words: Energy efficiency, HPC Challenge, Exaflop, power-performance profiling, power consumption

The International Journal of High Performance Computing Applications,
Volume 23, No. 3, Fall 2009, pp. 000–000
DOI: 10.1177/1094342009106193
© 2009 SAGE Publications Los Angeles, London, New Delhi and Singapore
Figures 1–7 appear in color online: http://hpc.sagepub.com

1 Introduction

Today it is not uncommon for large-scale computing systems and data centers to consume massive amounts of energy, typically 5–10 megawatts in powering and cooling (Cameron et al., 2005). This results in large electricity bills and reduced system reliability due to increased heat emissions. The continuing need to increase the size and scale of these data centers means that energy consumption potentially limits future deployments of high performance computing (HPC) systems.

Energy efficiency is now a critical consideration for evaluating high performance computing systems. For example, the Green500 list (Feng and Cameron, 2007) was launched in November 2006 to rank the energy efficiency of worldwide supercomputer installations. Meanwhile, the TOP500 supercomputing list, which ranks the most powerful supercomputing installations, has also started to report total power consumption of its supercomputers (University of Tennessee, 2008).

Today, there is no clear consensus as to which benchmark is most appropriate for evaluating the energy efficiency of high performance systems. Both the Green500 and the Top500 use the LINPACK (Dongarra, 1990) benchmark in their evaluations. In both cases, LINPACK was chosen more for reporting popularity than for its ability to evaluate energy efficiency. As with most benchmark activities, many high performance computing stakeholders do not agree with the choice of LINPACK as an energy efficiency benchmark.

Despite the increasing importance of energy efficiency and the controversy surrounding the need for standardized metrics, there have been few studies detailing the energy efficiency of high performance applications. Understanding where and how power is consumed for benchmarks used in acquisition is the first step towards effectively tracking energy efficiency in high performance systems and ultimately determining the most appropriate benchmark. The only detailed HPC energy efficiency study to

[1] SCAPE LABORATORY, VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY, BLACKSBURG, VA 24060, USA
(S562673@VT.EDU)

[2] MARQUETTE UNIVERSITY, MILWAULKEE, WI 53233, USA
(RONG.GE@MARQUETTE.EDU)

[3] VIRGINIA BIOINFORMATICS INSTITUTE, VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY, BLACKSBURG, VA 24060, USA
(FENGX@VBI.VT.EDU)

[4] SCAPE LABORATORY, VIRGINIA POLYTECHNIC INSTITUTE AND STATE UNIVERSITY, BLACKSBURG, VA 24060, USA
(CAMERON@VT.EDU)

date has been for the National Aeronautics and Space (NAS) parallel benchmarks (Feng et al., 2005).

While both the NAS benchmarks and LINPACK are widely used, the HPC Challenge (HPCC) benchmarks (Luszczek et al., 2005) were specifically designed to cover aspects of application and system design ignored by the former benchmarks and to aid in system procurements and evaluations. HPCC is a benchmark suite developed by the Defense Advanced Research Projects Agency (DARPA) High Productivity Computing System (HPCS) program. HPCC consists of seven benchmarks; each focuses on a different part of the extended memory hierarchy. The HPCC benchmark provides a comprehensive view of a system's performance bounds for real applications.

A detailed study of the energy efficiency of the HPCC benchmarks is crucial to understanding the energy boundaries and trends of HPC systems. In this paper, we present an in-depth experimental study of the power and energy profiles for the HPC Challenge benchmarks on real systems. We use the PowerPack toolkit (Feng et al., 2005; Cameron et al., 2005) for power-performance profiling and data analysis at function and component level in a parallel system. This work results in several new findings:

- The HPCC benchmark not only provides performance bounds for real applications, but also provides boundaries and reveals some general trends for application energy consumption.
- Each HPCC application has its own unique power profiles and such profiles vary with runtime settings and system configurations.
- An application's memory access rate and locality strongly correlate to power consumption. Higher memory access locality is indicative of higher power consumption.
- System idle power contributes to energy inefficiency. Lower system idling power would significantly increase energy efficiency for some applications.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 reviews PowerPack and the HPCC benchmarks. Sections 4 and 5 present an in-depth examination and analysis of the power and energy profiles of the HPCC benchmarks. Section 6 summarizes our conclusions and highlights future research directions.

2 Related Work

HPC researchers and users have begun measuring and reporting the energy efficiency of large-scale computer systems. Green500 (Feng and Cameron, 2007), TOP500 (University of Tennessee, 2008), and SPECPower (SPEC, 2008) provide methodologies and public repositories for tracking total system-wide power and efficiency. HPC systems contain diverse numbers and types of components including processors, memory, disks, and network cards at extreme scales. Understanding the energy scalability of an HPC system thus requires tracking of the energy efficiency of the entire system, as well as each component within the system at scale. However, all of the aforementioned methodologies and lists consider only the aggregate power of an entire system, and lack critical details as to the proportional energy consumption of individual components.

Most early power measurement studies focused on system- or building-level power (LBNL, 2003) measured with proprietary hardware (IBM, 2007), through power panels, or empirical estimations using rules of thumb (Bailey, 2002). The first detailed energy study of HPC applications (Feng et al., 2005) confirmed that power profiles of the NAS parallel benchmarks at the system level corresponded to performance-related activity such as memory accesses. These results were later confirmed in a study (Shoaib, 2008) that additionally measured high performance LINPACK (HPL) on a number of systems.

In previous work (Feng et al., 2005), we developed a software/hardware toolkit called PowerPack and used it to evaluate the power profiles and energy efficiency for various benchmarks, including the entire NPB benchmark suite on several high-performance clusters. PowerPack provides the infrastructure and methodology for power and performance data logging, correlation of data to source code, and post processing of the data collected for the benchmark under study. In addition to the software framework, PowerPack requires power measurement hardware to obtain direct measurements of the entire system and each component.

We use a direct measurement methodology for capturing performance and power data. There have been other efforts to capture HPC energy efficiency analytically (Ge and Cameron, 2007; Cho and Melhem, 2008; Yang Ding et al., 2008). For example, the Power-aware Speedup model was proposed (Ge and Cameron, 2007) to generalize Amdahl's law for energy. Another approach was proposed to study the energy implications of multi-core architectures (Cho and Melhem, 2008). PowerPack provides direct measurement, since we are interested in profiling the impact of energy consumption on HPC systems and applications. These analytical approaches are appropriate for further analyses using the data provided by tools such as PowerPack.

In this work, we study component- and function-level power consumption details for the HPCC benchmarks on advanced multi-core architectures. The HPCC benchmarks were chosen since they were designed to explore the performance boundaries of current systems. Some results, such as the high-performance LINPACK portion of the HPCC suite, have appeared in aggregate total-system form in the literature, but not at function- and component-level granularity. Other portions of the suite, such as STREAM, have not been profiled for power previously at this level of detail, to the best of our knowledge. Since the HPCC benchmarks collectively test the critical performance aspects of a machine, we study in this work whether these benchmarks also examine the critical energy aspects of a system.

3 Descriptions of the Experimental Environment

To detail the power/energy characteristics of the HPCC benchmarks on high performance systems, we use experimental system investigation. We begin by describing the setup of our experimental environment, which consists of three parts: the PowerPack power-profiling framework, the HPCC benchmark suite, and the HPC system under test.

3.1 The PowerPack Power-Profiling Framework

PowerPack is a framework for profiling power and energy of parallel applications and systems. It consists of both hardware for direct power measurement, and software for data collection and processing. PowerPack has three critical features: 1) direct measurements of the power consumption of a system's major components (i.e. CPU, memory, and disk) and/or an entire computing unit (i.e. an entire compute node); 2) automatic logging of power profiles and synchronization to application source code; and 3) scalability to large-scale parallel systems. PowerPack uses direct measurements from power meters to obtain accurate and reliable power and energy data for parallel applications. The current version of PowerPack supports multi-core multi-processor systems with dynamic power management configurations.

Figure 1 shows the software architecture of the PowerPack framework. PowerPack consists of five components: hardware power/energy profiling, software power/energy profiling control, software system power/energy control, data collection/fusion/analysis, and the system under test. The hardware power/energy profiling module simultaneously measures system power and multiple components' power. PowerPack measures the system power using a power meter plugged between the computer power supply units and electrical outlets. Component power is measured using precision resistors inserted into the DC lines carrying output from the computer's power supply unit, and voltage meters attached at two ends of the resistors. The software power/energy profiling control module provides the meter drivers and interface (i.e. application programming interfaces (APIs)) for the system or an application to start, stop, or label power profiling. The

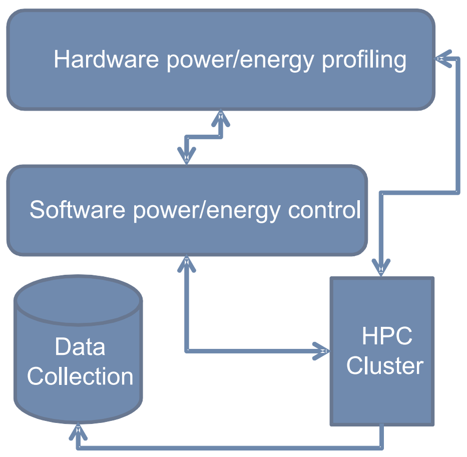

Hardware power/energy profiling
Software power/energy control
Data Collection
HPC Cluster

Fig. 1 PowerPack software architecture.

software system power/energy module is a set of interfaces (i.e. APIs) to turn on or off, or to set the frequencies, of processors. The data fusion module merges multiple data streams and transforms them into a well-structured format. Normally PowerPack directly measures one physical node at a time, but it is scalable to the entire computer cluster by using node remapping and performance-counter-based analytical power estimations.

3.2 The HPC Challenge Benchmark

The HPC Challenge (HPCC) benchmark (Luszczek, 2005) is a benchmark suite that aims to augment the TOP500 list and to evaluate the performance of HPC architectures from multiple aspects. HPCC organizes the benchmarks into four categories; each category represents a type of memory access pattern characterized by the benchmark's memory access spatial and temporal locality. Currently, HPCC consists of seven benchmarks: HPL, STREAM, RandomAccess, PTRANS, FFT, DGEMM and b_eff Latency/Bandwidth. To better analyze the power/energy profiles in the next section, we summarize the memory access patterns and performance bounds of these benchmarks in Table 1.

We use a classification scheme to separate the distinct performance phases that make up the HPCC benchmark suite. In *Local* mode, a single processor runs the benchmark. In *Star* mode, all processors run separate independent copies of the benchmark with no communication. In *Global* mode, all processing elements compute the benchmark in parallel using explicit data communications.

Table 1
Performance characteristics of the HPCC benchmark suite.

Benchmark	Spatial locality	Temporal locality	Mode	Detail
HPL	high	high	Global	Stresses floating-point performance
DGEMM	high	high	Star + Local	Stresses floating-point performance
STREAM	high	low	Star	Measures memory bandwidth
PTRANS	high	low	Global	Measures data transfer rate
FFT	low	high	Global + Star + Local	Measures floating-point and memory-transfer performance
RandomAccess	low	low	Global + Star + Local	Measures random updates of integer memory
Latency/Bandwidth	low	low	Global	Measures latency and bandwidth of communication patterns

3.3 The HPC System Under Test

The Dori system is a cluster of nine two-way dual-core AMD Opteron 265-based server nodes. Each node consists of six 1-GB memory modules, one Western Digital WD800 SATA hard drive, one Tyan Thunder S2882 motherboard, two CPU fans, two system fans, three Gigabit Ethernet ports, and one Myrinet interface. The compute nodes run CentOS Linux with kernel version 2.6. During our tests, we used MPICH2 as the communication middleware. The network interconnection is via gigabit Ethernet.

4 HPCC Benchmark Power Profiling and Analysis

4.1 A Snapshot of the HPCC Power Profiles

The power profiles of each application are unique. Figure 2 shows power profiles of the HPCC benchmarks running on two nodes of the Dori cluster. These profiles are obtained using the problem size where HPL achieves its maximum performance on two nodes. Power consumption is tracked for major computing components including CPU, memory, disk and motherboard. As we will see later, these four components capture nearly all the dynamic power usage of the system that is dependent on the application.

A single HPCC test run consists of a sequence of eight benchmark tests as follows: 1) Global PTRANS; 2) Global HPL; 3) Star DGEMM + Local DGEMM; 4) Star STREAM; 5) Global MPI RandomAccess; 6) Star RandomAccess; 7) Local RandomAccess; and 8) Global MPI FFT, Star FFT, Local FFT and Latency/Bandwidth. Figure 2 clearly shows the unique power signature of each mode of the suite. We make the following observations from Figure 2:

- Each test in the benchmark suite stresses processor and memory power relative to their use. For example, as Global HPL and Star DGEMM have high temporal and spatial locality, they spend little time waiting on data, stress the processor's floating point execution units intensively, and consume more processor power than other tests. In contrast, Global MPI RandomAccess has low temporal and spatial memory access locality; thus this test consumes less processor power due to more memory access delay, and more memory power due to cache misses.
- Changes in processor and memory power profiles correlate with communication to computation ratios. Power varies for global tests such as PTRAN, HPL, and MPI_FFT because of their computation and communication phases. For example, the HPL computation phases run at 50 watts higher than its communication phases. Processor power does not vary as much during Star (embarrassingly parallel) and Local (sequential) tests due to limited processing variability in the code running on each core. In Global modes, memory power varies, but not widely in absolute wattage since memory power is substantially less than processor power on the system under test.
- Disk power and motherboard power are relatively stable over all tests in the benchmarks. None of the HPCC benchmarks stresses local disks heavily. Thus power variations due to disk use are not substantial. On this system, the network interface card (NIC), and thus its power consumption, is integrated in the motherboard. Nonetheless, communication using the gigabit Ethernet card does not result in significant

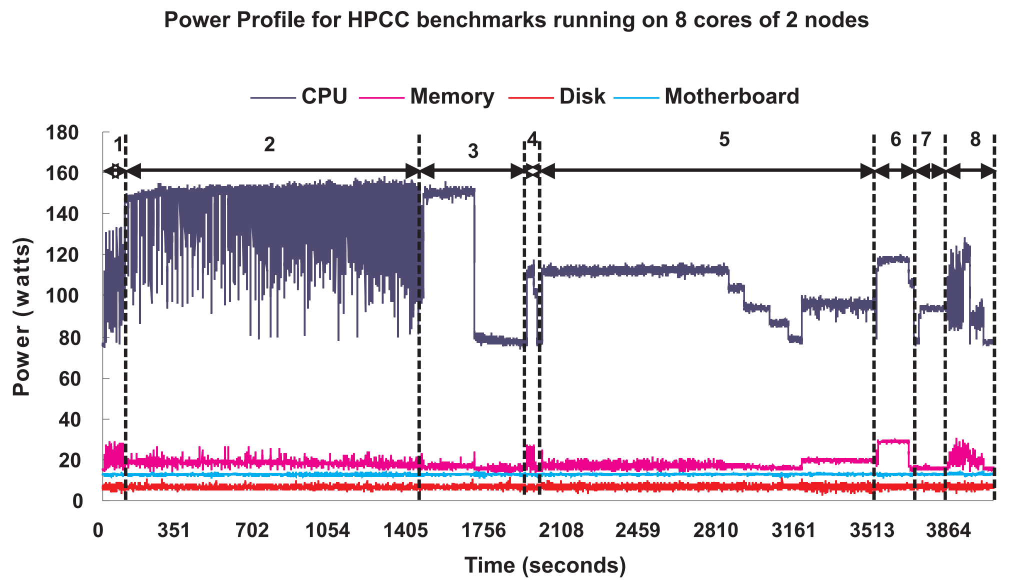


Fig. 2 A snapshot of the HPCC power profile. The power consumption of major components per compute node when running a full HPCC benchmark suite using eight cores. The entire run of HPCC consists of eight micro benchmark tests in this order: 1) PTRANS; 2) HPL; 3) Star DGEMM + single DGEMM; 4) Star STREAM; 5) MPI RandomAccess; 6) Star RandomAccess; 7) Single RandomAccess; 8) MPI FFT, Star FFT, single FFT and latency/bandwidth. In this test, the problem size fits the peak execution rate for HPL. For LOCAL tests, we run a benchmark on a single core with three idle cores.

power use even under the most intensive communication phases.

- Processors consume more power during Global and Star tests since they use all processor cores in the computation. Local tests use only one core per node and thus consume less energy.

4.2 Power Distribution Among System Components

Each test within the HPCC benchmarks is unique in its memory access pattern. Together, they collectively examine and identify the performance boundaries of a system with various combinations of high and low temporal and spatial locality, as described in Table 1. We hypothesize that the HPCC benchmarks will identify analogous boundaries for power and energy in high performance systems and applications. HPL (high, high), PTRAN (low, high), Star_FFT (high, low) and Star_RandomAccess (low, low) represent the four combinations of (temporal, spatial) locality captured by the HPCC tests. Figure 3 superimposes the results for these four tests on an X–Y plot of temporal–spatial sensitivity. For comparison, below the graph we also provide the power breakdowns for system idle (i.e. no active user applications), and for disk copy: this stresses disk I/O access, which is not captured by the HPCC tests.

The total power consumption figures for each of the isolated phases of the HPCC benchmark depicted in Figure 3, in increasing order, are: system idle (157 watts), disk copy (177 watts), Star_RandomAccess (219 watts), Star_FFT (221 watts), PTRAN (237 watts), and HPL (249 watts). The power numbers for the two Star benchmarks are representative given the stable power profiles, while the power numbers for PTRAN and HPL are of time instances when processor power reaches the respective maximum. We conclude the following from the data:

- Since lower temporal and spatial locality imply higher average memory access delay times, applications with (low, low) temporal–spatial locality use less power on average. Memory and processor power dominate the power budget for these tests. Idling of the processor during memory access delays results in more time spent idling at lower dynamic processor power use, thus leading to lower average system power use.
- Since higher temporal and spatial locality imply lower average memory access delay times, applications with (high, high) temporal–spatial locality use more power on average. Memory and processor power dominate

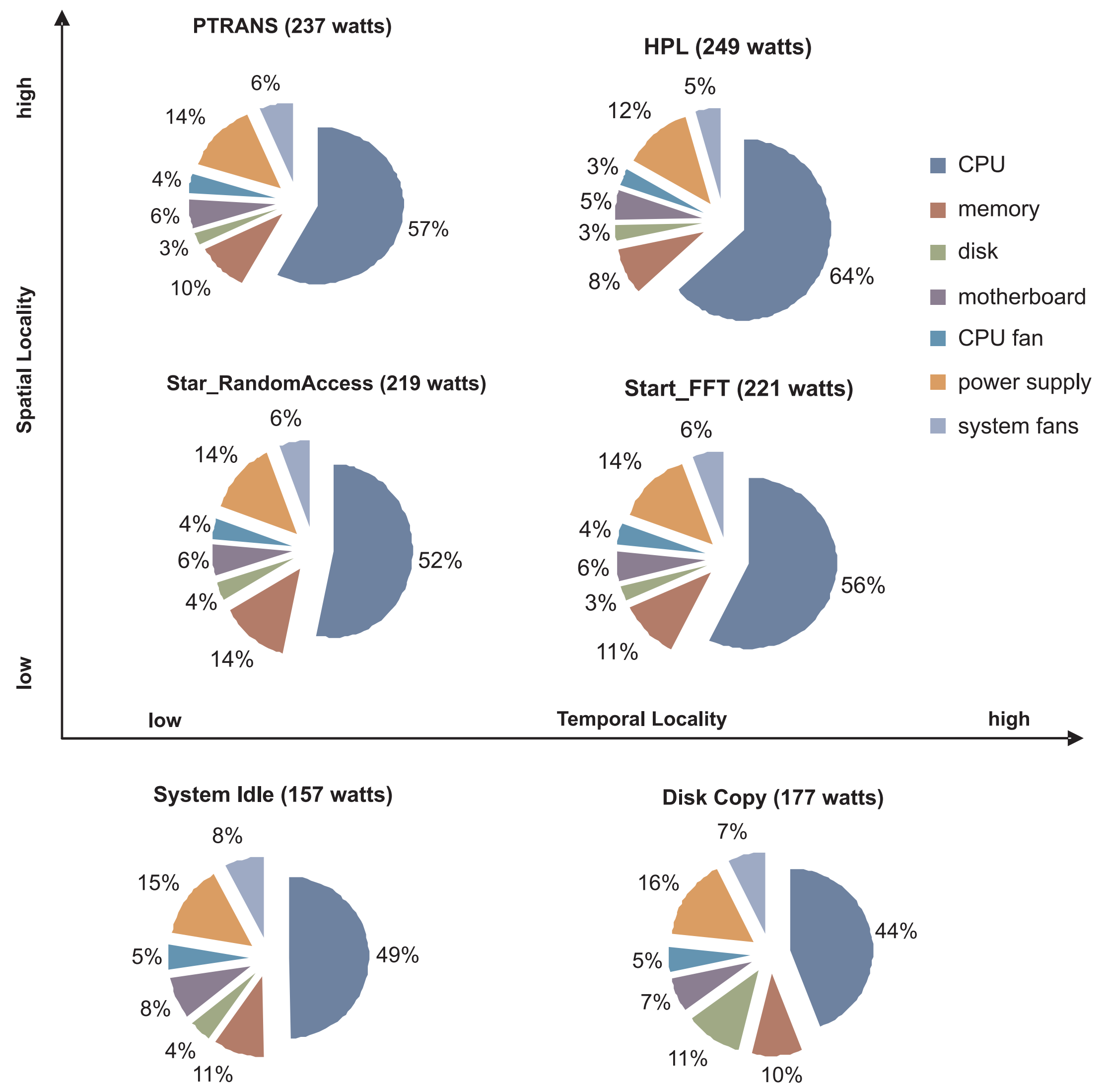


Fig. 3 System power distribution for different workload categories. This figure shows system power broken down by component for six workload categories. The top four categories represent four different memory access locality patterns. The other two workloads are system idle, i.e., no user application running, and disk copy, i.e., concurrently running the Linux standard copy program on all cores. The number given in the subheading for each category is its total average power consumption.

the power budget for these tests. Intensive floating-point execution leads to more activities, and higher active processor power and average system power. Since HPL has both high spatial and high temporal locality, it has the highest power consumption and thus is a candidate for measuring the maximum system power that an application would require.

- Mixed temporal and spatial locality implies mixed results that fall between the average power ranges of (high, high) and (low, low) temporal–spatial locality tests. These tests must be analyzed more carefully by phase, since communication phases can affect the power profiles substantially.

Overall, processor power dominates total system power for each test shown in Figure 3; the processor accounts for as little as 44% of the power budget for disk copy and as much as 61% of the power budget for HPL. Processor power also shows the largest variance: as high as 59 watts for HPL. For workloads with low temporal and spatial locality, such as Star_RandomAccess, processor power consumption varies by as much as 40 watts.

Memory is the second largest power consumer in the HPCC tests and accounts for 8% (HPL) and 13% (RandomAccess) of the consumption of the total system. Low spatial locality causes additional memory accesses and thus higher memory power consumption. For Star_RandomAccess, the memory power consumption increases by 11 watts over that of the idle system.

Disk normally consumes much less power than processor and memory, but may consume up to 11% of total system power for disk-I/O-intensive applications such as disk copy. HPCC, LINPACK, and the NAS parallel benchmarks, like most HPC benchmarks, do not stress disk accesses. This may be due to the fact that in high performance environments with large storage needs the data storage is remote from the computational nodes, in a disk array or other data appliance. We use the disk copy benchmark primarily to illustrate the way the power budget can change under certain types of loads; this means that local disk power should not be readily dismissed for real applications.

The system under test consumes a considerable amount of power when there is no workload running. We have omitted discussion of power supply inefficiency since it is fundamentally an electrical engineering problem, which accounts for another 20–30% of the total system power consumed under these workloads. Altogether, the processor, the power supply, and the memory are the top three power-consuming components for the HPCC benchmarks and typically account for about 70% of total system power.

4.3 Detailed Power Profiling and Analysis of Global HPCC Tests

The previous subsection focused on the effect of memory access localities of applications on power distribution over the system components. In this subsection we study the way in which parallel computation changes the locality of data accesses and impacts the major computing components' power profiles over the execution of the benchmarks. We still use the same four benchmarks, all of them in Global mode.

PTRANS implements a parallel matrix transposition. HPCC uses PTRANS to test system communication capacity using simultaneous large message exchanges between pairs of processors. Figure 4(a) shows the power profiles of PTRANS with problem size $N = 15\,000$. Processor power varies within iterations. The spikes correspond to computation phases and valleys correspond to system-wide large message exchanges, which occur on iteration boundaries as marked in the figure. While PTRANS's power consumption is reasonably high during computation phases, its power consumption during communication phases is close to the lowest. Also, the memory power profile is stable with only slight changes. PTRANS has high spatial locality and consequently a low cache miss rate. Communication phases are sufficiently long for the processor to reduce its power consumption. Additionally, the computation phases of PTRANS focus on data movement to perform the transposition, which is less computationally intensive than the matrix computations in HPL; this results in a lower average power consumption for the PTRANS computation phases.

HPL is a computation-intensive workload interspersed with short communications. HPCC uses HPL to assess peak floating-point performance. HPL's processor power profiles display high variance, as shown in Figure 4(b). For this data, we experimentally determined a problem size where HPL achieves peak execution rate on 32 cores across eight nodes. At peak HPL performance, processor power reaches about 150 watts, which is the maximum over all HPCC tests. Nevertheless, processor power fluctuates dramatically, and it may drop to 90 watts during the short communications in HPL. Memory power does not vary much during HPL execution. This is because HPL has high temporal and spatial memory access locality, and thus does not frequently incur cache misses.

MPI_RandomAccess is a Global test that measures the rate of integer updates to random remote memory locations. This code's memory access pattern has low spatial and temporal locality. As shown in Figure 4(c), the processor power profile for MPI_RandomAccess follows a steady step function that reflects the relatively low computational intensity of random index generation. MPI_RandomAccess stresses inter-processor communication of small messages. Processor power decreases during the message exchange of indices with other processors. Memory power jumps towards the end of the run as the local memory updates occur once all indices are received. This memory power consumption is notably almost the highest among all the Global tests. The only higher memory power consumption we observed was for the embarrassingly parallel (EP) version of this code, Star_RandomAccess.

MPI_FFT is a Global test that performs the fast Fourier transform operation. Figure 4(d) shows the power profiles of MPI_FFT with input vector size of 33 554 432. MPI_FFT's processor power profile is similar to that of PTRANS, in which communication tends to decrease processor power on function boundaries, as marked in the figure. Unlike that of PTRANS, however, the memory power profile of MPI_FFT exhibits obvious fluctuations. These differences are the results of different memory access patterns inherent in these two benchmarks. While

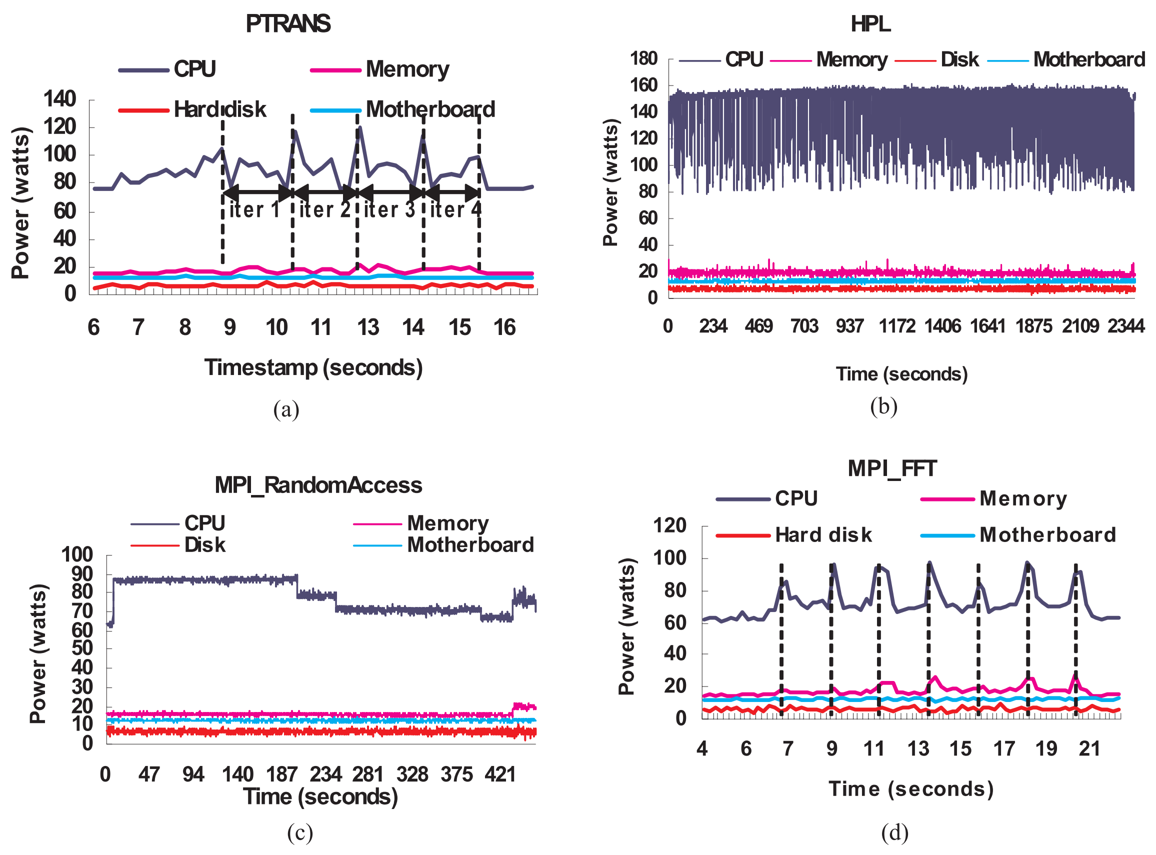


Fig. 4 Detailed power profiles of four HPCC benchmarks running across eight nodes on 32 cores in Global mode. This figure shows the power consumptions of system major components over time for a) PTRANS, b) HPL, c) MPI_RandomAccess, and d) MPI_FFT during their entire execution.

PTRANS has low temporal locality and high spatial locality, MPI_FFT has high temporal locality and low spatial locality. Higher miss rates lead to higher power consumption in memory for MPI_FFT compared with PTRANS.

Figure 5 shows the amplified version of MPI_FFT at the function level. FFT conducts global transpositions for large complex vectors via inter-processor communication and spends over 70% of its execution time in communication phases. The seven spikes in the CPU power profile correspond to seven computation-intensive phases: vector generation, computation1, computation2, random vector generation, inverse computation1, inverse computation2, and computation of error. The valleys in the processor power profile correspond to data transpositions involving inter-processor communications. The interesting observation is that memory power goes up during computation phases and drops during communication phases, meaning that memory access is more intensive for the former. Such profiling reveals the detailed dynamics of system activity and power consumption.

5 HPCC Benchmark Energy Profiling and Analysis

While we are interested in power (P), the rate of energy consumption at time instances when an application executes, we are also interested in the system energy consumption (E) required to finish executing the application. Energy is the area under the curves given by the power profiles previously discussed. It reflects the electrical cost of running an application on a particular system. In this section we analyze the resulting impact of power profiles on energy consumption for parallel systems. In particular, we study the energy efficiency of these applications and systems as they scale.

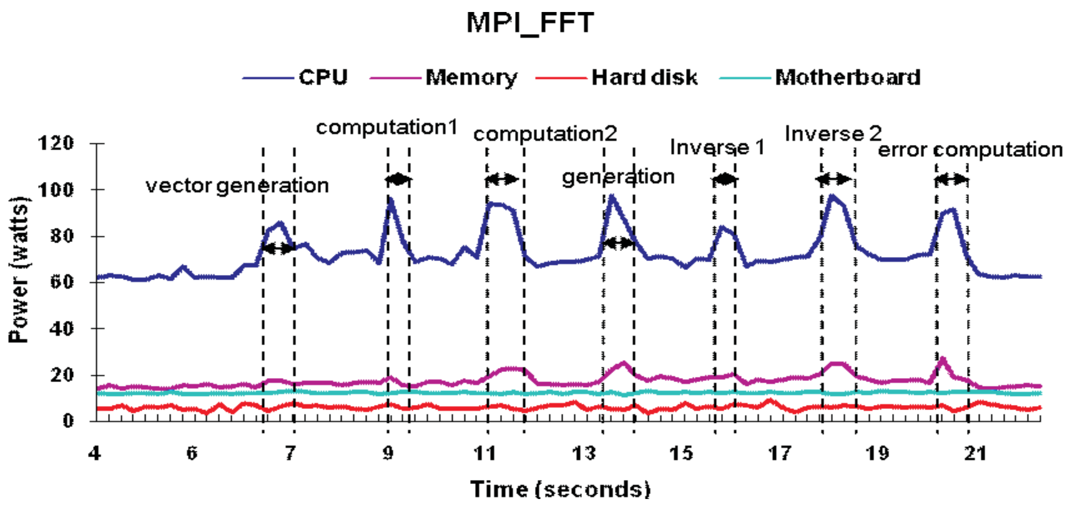


Fig. 5 Detailed power-function mapping of MPI_FFT in HPCC. PowerPack shows processor power rises during computation phases, and drops during communication phases. The seven spikes in processor power profile correspond to vector generation, computation1, computation2, random vector generation, inverse computation1, inverse computation2, and computation of error; the valleys correspond to transpositions that involve inter-processor communications.

5.1 Energy Profiling and Efficiency of MPI_FFT under Strong Scaling

Strong scaling describes the process of fixing a workload and scaling up the number of processors. Embarrassingly parallel codes often scale well under strong scaling conditions. For these types of codes, scaling system size reduces execution times linearly with constant nodal power consumption. As a result, the total energy consumption maintains the same. When measuring energy efficiency with the amount of workload computed per unit of energy, an embarrassingly parallel code achieves better energy efficiency as system size scales.

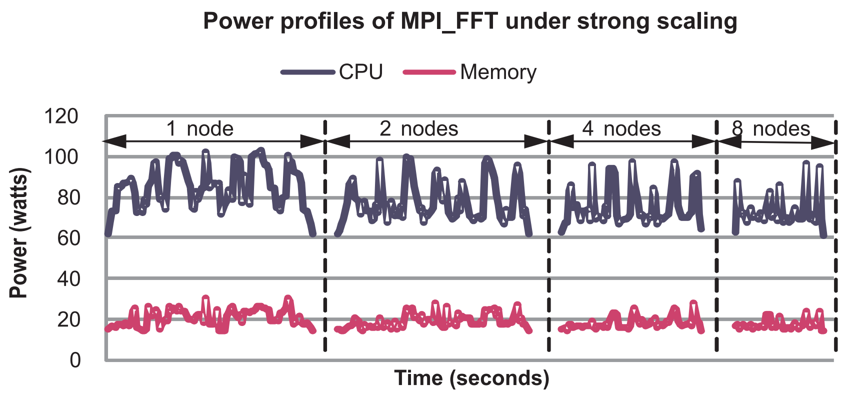
However, for codes that are not embarrassingly parallel, such as Global MPI_FFT, the energy efficiency is less clear. In Figure 6, we show the power per node, system energy, performance, and resulting energy efficiency of a set of Global MPI_FFT tests. Using a fixed problem size that is the same as the one mentioned in Section 4.3, we run MPI_FFT at different numbers of nodes. During the test, we use all the four cores on each node. Thus, running on eight nodes actually creates 32 threads with one thread per core. Figure 6(a) shows the components' power per node. As system size scales, the duration of high processor power phases (computation phases) becomes shorter, and the duration of lower processor power phases (communication phases) becomes longer. In addition, the highest processor power within one run slightly decreases when the number of nodes increases. Memory power also shows a similar trend.

Figure 6(b) shows that both performance and energy increase sub-linearly as system size scales. However, energy increases much faster than performance for the system sizes we have measured. For example, running on eight nodes gains a 2.3 times speedup, but costs 3.3 times as much energy as that used when running on one node. This observation implies the need for appropriate trade-offs between running an application faster and maintaining sufficient energy efficiency, or constraining running costs.

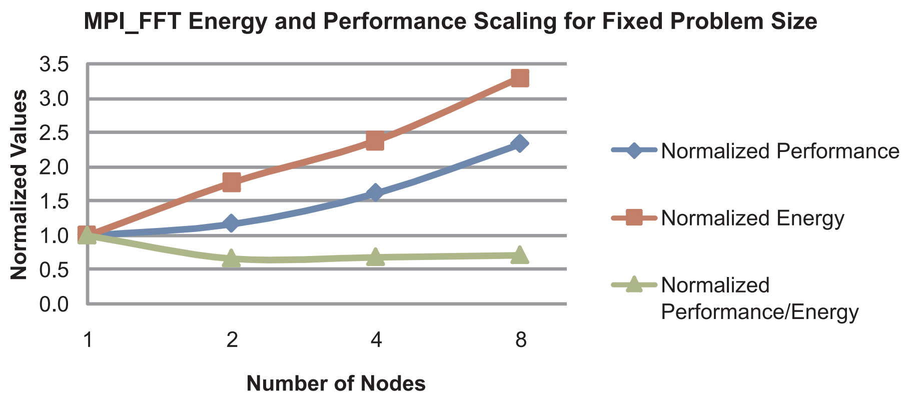
5.2 Energy Profiling and Efficiency of HPL under Weak Scaling

Weak scaling describes the process of simultaneously increasing both workload and the number of processors being used. The discussions on energy efficiency under strong scaling imply that in order to maintain a desired level of energy efficiency, we also need to maintain the parallel efficiency of the application. In the experiments below, we want to find out how effectively the weak scaling case can maintain its parallel efficiency and its energy efficiency.

Figure 7 shows the directly measured power, energy, performance, and energy efficiency of HPL on the Dori cluster under weak scaling. In our experiments we increase



(a)



(b)

Fig. 6 The power, energy, performance, and energy efficiency of MPI_FFT. In this figure we run the Global MPI_FFT test for a fixed problem size, using a varying number of nodes with four cores per node. Both performance and energy are normalized against the respective values when using one node. We use normalized performance/energy as a measure of achieved energy efficiency.

the problem size of HPL in such a way that HPL always achieves its maximum performance on the respective number of nodes. Figure 7(a) shows the power profiles of HPL. With weak scaling, neither processor power nor memory power change perceptibly with the number of nodes during the computation phases, though power decreases slightly during the communication phases as the system scales. As a result, average nodal power remains almost the same.

Figure 7(b) shows the normalized performance and derived total energy. We observe that both performance and energy increase roughly linearly as system size increases. Accordingly, the energy efficiency remains constant as system size increases. These observations exactly match our expectations. By increasing the problem size of HPL to match the system's maximum computational power, we maintain the memory spatial and temporal locality and the parallel efficiency, which in turn leads to constant energy efficiency. We would like to note that observations from weak scaling of HPL are very close to those from the embarrassingly parallel case.

6 Summary and Conclusions

In summary, we evaluated the power and performance profiles of the HPC Challenge benchmark suite on a multi-core-based HPC cluster. We used the PowerPack toolkit to collect the power and energy profiles and isolate them by components. We organized and analyzed the power and energy profiles according to each benchmark's memory access locality.

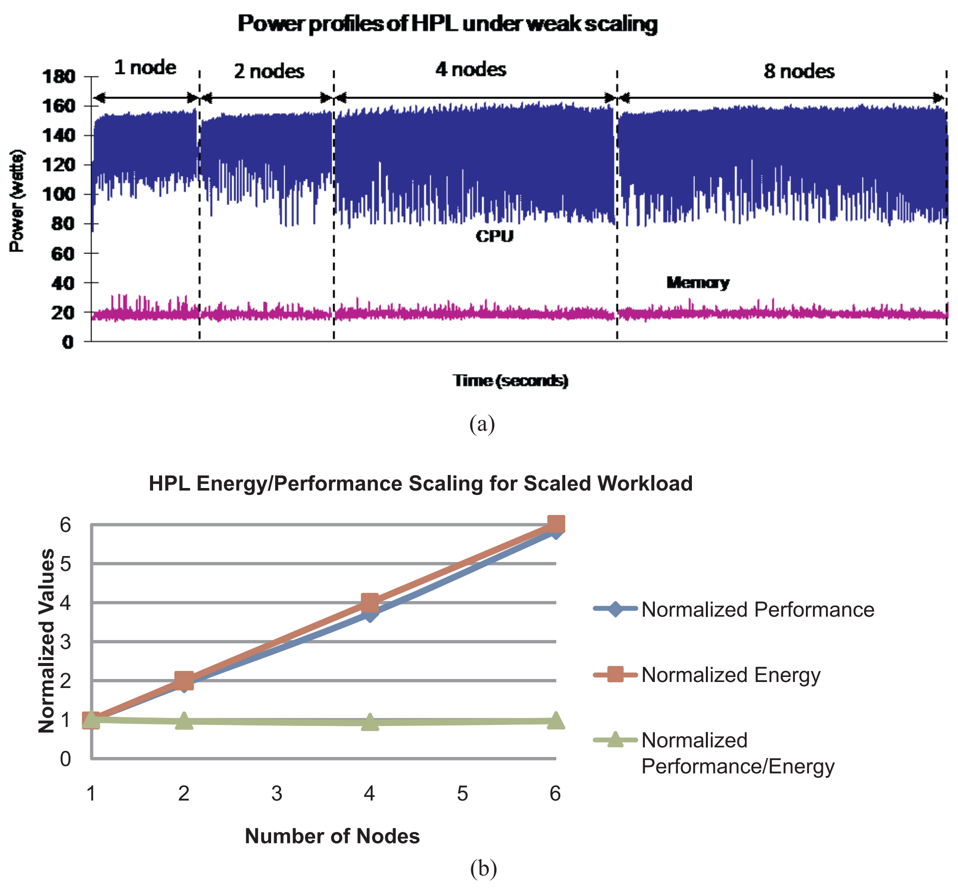


Fig. 7 The power, energy, performance, and energy efficiency of HPL under weak scaling. In this figure, both performance and energy are normalized against the corresponding values when using one node. We also plot the normalized performance/energy value as a measure of achieved energy efficiency.

Each application has a unique power profile characterized by power distribution among major system components, especially processor and memory, as well as power variations over time. By correlating the HPCC benchmark's power profiles to performance phases, we directly observed strong correlations between power profiles and memory locality.

The power profiles of the HPCC benchmark suite reveal power boundaries for real applications. Applications with high spatial and temporal memory access locality consume the most power but achieve the highest energy efficiency. Applications with low spatial or temporal locality usually consume less power but also achieve lower energy efficiency.

Energy efficiency is a critical issue in high performance computing, and it requires further study, since the interactions between hardware and application affect power usage dramatically. For example, for the same problem on the same HPC system, choosing an appropriate system size will significantly affect the achieved energy efficiency.

In the future, we will extend the methodologies used in this paper to larger systems and novel architectures. Additionally, we would like to explore further energy-efficiency models and the insight they provide to the sustainability of large-scale systems.

Author's Biography

Shuaiwen Song is a PhD candidate in the Department of Computer Science and a researcher at the Scape Laboratory at Virginia Polytechnic Institute and State University. His research interests include parallel and distributed systems, multi-core systems, power-aware and high performance computing, performance modeling and analysis. Shuaiwen received his BE in Software Engineering from Dalian University of Technology, China. He is a student member of IEEE.

Rong Ge is an Assistant Professor of Computer Science at Marquette University. She received the PhD degree in Computer Science from Virginia Polytechnic Institute and State University. Her research interests include performance modeling and analysis, parallel and distributed systems, energy-efficient computing, high performance computing, and computational science. She is a member of the IEEE, the IEEE Computer Society, the ACM and Upsilon Pi Epsilon.

Xizhou Feng is a Senior Research Associate in the Network Dynamics and Simulation Science Laboratory at Virginia Polytechnic Institute and State University. He received the PhD degree in Computer Science from the University of South Carolina. His research interests include high performance computing algorithms and systems, distributed systems and cyber-infrastructure, computational biology and bioinformatics, and modeling and simulation of complex systems. He is a member of the IEEE, the IEEE Computer Society and the ACM.

Kirk W. Cameron is an Associate Professor of Computer Science at Virginia Polytechnic Institute and State University. Professor Cameron received the B.S. in Mathematics from University of Florida in 1994 and the Ph.D. in Computer Science from Louisiana State University in 2000. He directs the SCAPE Laboratory at Virginia Tech where he pioneered the area of high-performance, power-aware computing to improve the efficiency of high-end systems. Cameron has received numerous awards and accolades for his research and publications including the National Science Foundation Career Award (2004), the department of energy Career Award (2004), University of South Carolina College of Engineering Young Investigator Research Award (2005), Best Paper Nominee SC06, Virginia Polytechnic Institute and State University College of Engineering Fellow (2007) and the IBM Faculty Award (2007). He is an Uptime Institute Fellow (2008), and was invited to the 2008 National Academy of Engineering Symposium. Professor Cameron is on the editorial board and editor for the *IEEE Computer* "Green IT" column.

References

Bailey, A..M. 2002. Accelerated Strategic Computing Initiative (ASCI): Driving the need for the Terascale Simulation Facility (TSF). In *Energy 2002 Workshop and Exposition*, Palm Springs, CA.

Cameron, K.W., Ge, R., and Feng, X. 2005. High-performance, power-aware distributed computing for scientific applications. *IEEE Computer* **38**(11): 40–47.

Cho, S. and Melhem, R. 2008. Corollaries to Amdahl's Law for energy. *Computer Architecture Letters* **7**(1): 25–28.

Dongarra, J. 1990. The LINPACK Benchmark: an explanation. *Evaluating Supercomputers: Strategies for Exploiting, Evaluating and Benchmarking Computers with Advanced Architectures*, Chapman & Hall, Ltd., London, pp. 1–21.

Feng, W.-C. and Cameron, K.W. 2007. The Green500 List: encouraging sustainable supercomputing. *IEEE Computer* **40**(12): 50–55.

Feng, X., Ge, R. and Cameron, K.W. 2005. Power and Energy Profiling of Scientific Applications on Distributed Systems (IPDPS 05). In *19th IEEE International Parallel and Distributed Processing Symposium,* Denver, CO.

Ge, R. and Cameron, K. 2007. Power-aware speedup. In *Parallel and Distributed Processing Symposium*, IPDPS.

IBM, 2007. *PowerExecutive*. Available at http://www-03.ibm.com/systems/management/director/extensions/powerexec.html.

LBNL, 2003. *Data Center Energy Benchmarking Case Study: Data Center Facility 5.* 2003.

Luszczek, P., Dongarra J, Koester D, Rabenseifner B, Lucas J, Kepner J, McCalpin J, Bailey D and Takahashi D. 2005. *Introduction to the HPC Challenge Benchmark Suite.* Seattle, WA, http://icl.cs.utk.edu/hpcc/pubs March 2005.

Shoaib, K., Shalf, J. and Strohmaier, E. 2008. Power efficiency in high performance computing. In *High-Performance, Power-aware Computing Workshop*, Miami, FL.

SPEC, 2008. *The SPEC Power Benchmark*. Available at http://www.spec.org/power_ssj2008/.

University of Tennessee, 2008. *Top 500 Supercomputer List*. SC08 Conference, Austin, TX, 15–21 November, 2008. Available at http://www.top500.org/list/2008/11/100.

Yang Ding, Malkowski, K., Raghavan, P. and Kandemir, M. 2008. Towards energy efficient scaling of scientific codes. *Parallel and Distributed Processing*, 2008. IPDPS 2008. IEEE International Symposium, April 2008 pp1–8.