Evaluation of Parallel I/O Performance and Energy with Frequency Scaling on Cray XC30

Suren Byna and Brian Austin

Lawrence Berkeley National Laboratory
Berkeley, CA 94720, USA.
Email: {sbyna, baustin}@lbl.gov

***Abstract*—Large-scale simulations produce massive data that needs to be stored on parallel file systems. The simulations use parallel I/O to write data into file systems, such as Lustre. Since writing data to disks is often a synchronous operation, the application-level computing workload on CPU cores is minimal during I/O and hence we consider whether energy may be saved by keeping the cores in lower power states. To examine this postulation, we have conducted a thorough evaluation of energy consumption and performance of various I/O kernels from real simulations on a Cray XC30 supercomputer, named Edison, at the National Energy Research Supercomputing Center (NERSC). To adjust CPU power consumption, we use the frequency scaling capabilities provided by the Cray power management and monitoring tools. In this paper, we present our initial observations that when the I/O load is high enough to saturate the capability of the filesystem, down-scaling the CPU frequency on compute nodes reduces energy consumption without diminishing I/O performance.**

I. Introduction

Power efficiency and energy efficiency are increasingly prominent concerns to the high-performance computing (HPC) community with upcoming exascale systems. As supercomputing systems approach the exascale regime, their power consumption (and associated infrastructure and operating costs) will grow proportionally unless their efficiency increases. The U.S. Department of Energy has set an ambitious efficiency goals of approximately 20 MW for future exascale systems. Thus, energy efficiency is becoming a first-order constraint for the design of future HPC systems. A recent DOE report states that energy efficiency is one of the most difficult challenges at exascale [4].

While flops per watt may be the top-billing metric for next generation systems, the performance of the growing of "big data" and data-intensive workloads will emphasize I/O performance more than flops. For example, plasma physics and cosmology codes simulate tens of trillions of particles and produce datasets in the range of tens to hundreds of terabytes per time step [31], [6]. The development of in-situ analysis and visualization tools (for example, in VisIt [7] or in ParaView[25]), is in part, a reflection of the potential imbalance between I/O and compute resources that are anticipated. File systems must also operate within the system's power budget, so the energy efficiency of I/O operations are critical.

There is a growing body of work studying application energy efficiency (see section II), but relatively limited investigation of parallel I/O energy efficiency. Ge et al. [15], [18] investigated the impact of application I/O patterns on energy consumption and proposed a middleware to apply dynamic voltage and frequency scaling (DVFS) for compute nodes. This study shows up to 9% to 28% energy reduction for I/O benchmarks that have contiguous or non-contiguous data reads/writes. However, this study was conducted on a small-scale dedicated cluster with few jobs running on the system. Supercomputing systems located at facilities such as the National Energy Research Supercomputing Center (NERSC) are shared by hundreds of users and the parallel file system is heavily used by a large number of applications. There has been no study to analyze the benefits of DVFS during I/O phases of applications on large-scale systems with applications that write terabytes of data.

In this paper, we present our initial observations of energy consumption and parallel I/O performance trade-offs with two I/O kernels and scaling them up to 16K cores and writing data up to 4 TB and 12 TB in size, respectively. We extracted the I/O kernels from real applications: plasma physics (Vector particle-in-cell - VPIC), and accelerator physics (VORPAL) simulations. The I/O kernel represents storing data to the file system at one time step and the real simulations typically store data at tens of such time steps.

The primary contributions of this paper are:

- We demonstrate that DVFS has potential to save energy and improve energy efficiency on large-scale systems for two I/O benchmarks sampled from real scientific applications.
- We show a collection of qualitative power/performance trade-off curves for the two benchmarks. We use "I/O rate per Watt" as a metric of energy efficiency.
- Our observations show that energy savings and energy efficiency differ for the two kernels we tested. The dependency of savings on I/O patterns needs to be studied further.

The remainder of this paper is organized as follows. Section II summarizes related work. Section III describes, at a high level, the factors that contributed to our experimental design. Section IV details the computational platform, I/O benchmarks and power measurement techniques used. Our results are presented in section V. We conclude in Section VI.

II. Related Work

A. Frequency scaling and energy efficiency

A significant number of research efforts have focused on saving power and energy of high-performance computing systems by taking advantage of frequency scaling capability of modern processors. A non-exhaustive list of these efforts includes [14], [33], [26], [16], [27], [22], [17], [23], [20]. Hsu et al. [22], [23] proved the feasibility of using DVFS (Dynamic Voltage and Frequency Scaling) to reduce processor power consumption. Freeh et al. [14] studied energy and execution time tradeoffs in MPI programs. Song et al. [33] proposed an analytical model to predict performance and energy consumption of applications based on various system and application parameters to help users balance energy use and performance. An Energy-aware Distributed Runtime (EDR) has been proposed by Li et al. [26] to efficiently select data replicas in data center environments. All these efforts explore saving power without impacting performance of computation and communication phases of applications. In this paper, we study the impact of power-saving strategies on performance during parallel I/O phases.

Various efforts focused on DVFS for parallel task scheduling on clusters [38], [37], [28], [19], [36]. Yao et al. [38] and Manzak et al. [28] proposed scheduling independent tasks with DVFS on a single processor systems. Wei et al. [37] and Gruian et al. [19] discuss scheduling dependent tasks on multiple processors using DVFS. Wang et al. [36] recently proposed a power-aware scheduling based on task clustering for dependent tasks by zeroing communication links to reduce power consumption. All these efforts aimed to reduce power consumption during computation and communication phases and none of them target the phases that move data to storage media.

B. Storage energy efficiency

Studies of energy efficiency of storage systems have focused on reducing power consumption of disks and I/O servers by keeping them in low power states. Multi-speed disks have been proposed to reduce energy dissipation in hard drives. Son et al. [32] proposed software-directed disk power management with energy-aware compilers for multi-speed disks. Several proposals focused on scheduling the I/O requests of applications and reducing power consumption of the I/O subsystem by placing some disks in low power states. Chou et al. [8], [9] and Kim et al. [24] propose strategies for energy-aware disk scheduling by analyzing the arrival patterns of I/O requests. All the above mentioned proposals used disk simulators and analytical models for evaluating power/energy consumption of parallel I/O systems. Moreover, suggestion of powering down or placing disks in lower power states is infeasible on large scale supercomputing systems, where hundreds of users are simultaneously using the I/O subsystem. Because the I/O subsystem is a shared resource, real-time analysis and scheduling of I/O requests would likely increase application overhead and lower energy efficiency. In contrast, this study focuses on the energy efficiency of compute nodes. Typically, compute nodes are dedicated to a single application and a batch processing service is used to assign compute nodes to an application.

Reducing parallel system energy consumption specifically during parallel I/O phases has been studied by Ge et al.. In [15], these authors investigate the impact of application's I/O access patterns, parallel file system deployment, and processor frequency scaling on energy efficiency; this study shows the feasibility of energy efficiency by processor frequency scaling. Their SERA-IO middleware package has been shown to save energy without effecting performance by intercepting MPI-I/O calls and interleaving DVFS commands [18]. These two studies were performed on a small-scale system in a dedicated environment. In this study, we investigate the impact of frequency scaling on compute nodes of a petascale Cray XC30 supercomputer, where the I/O subsystem is heavily shared by hundreds of simultaneous applications.

C. Using Cray PMDB

Measuring power and energy consumption on Cray XC30 systems has become possible with the Cray Power Management Database (PMDB) [29]. First experiences of these measurements were performed by Fourestey et al. [13] and Austin et al. [2]. While Fourestey et al. [13] validated the measurements of PMDB using various benchmarks, the latter study focused on evaluating first order energy and performance models using various compute intensive microbenchmarks. In this paper, we study power, energy, and performance impacts of I/O kernels retrieved from two highly scalable simulation codes from plasma physics and accelerator physics.

III. Motivation and Approach

The parallel I/O software stack contains multiple layers of software. As shown in Figure 1, scientific applications often use high-level I/O libraries, such as HDF5 [34] or NetCDF [35] to read or write data arrays to and from parallel file systems. These libraries internally use POSIX-IO and MPI-IO [10] middleware to perform I/O. I/O optimization layers, such as I/O Forwarding Layer [1] and the Scientific Data Services framework [11], perform optimizations such as data organization or redirection of I/O calls to different views of data to take advantage of parallelism available in file systems. All these different layers are interdependent to achieve efficient I/O performance.

Each of the parallel I/O software layers offers tunable parameters. For example, in the HDF5 layer, selecting chunking sizes to write multi-dimensional arrays improves future reads of the chunked data. MPI-IO offers two-phase I/O parameters that applications can use to reduce the number of readers or writers that interact directly with the file system. File systems, such as Lustre, allow users to set the number of storage targets to write data to and the size of contiguous chunks of data on each storage target. Optimizing I/O by selecting the right configuration of the above mentioned software layers improves

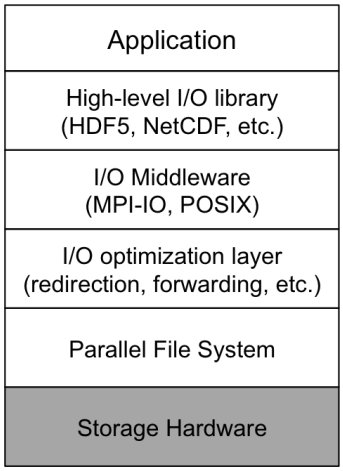

Fig. 1. Various layers of software in the contemporary parallel I/O stack

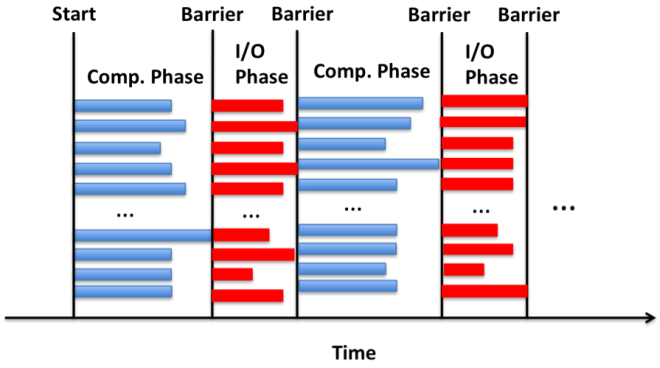

Fig. 2. Banded computation and I/O phases present in various parallel scientific simulations, where all MPI processes of a simulation perform computation and collectively perform I/O, i.e., the following computation phase starts after all the processes finish I/O.

I/O performance [3]. In our previous research, we have extensively studied tuning these parameters automatically.

Optimized configurations provide improved I/O performance that, in turn, results in less energy consumption. Since many scientific simulations perform computation and I/O in alternate phases (shown in Figure 2), we postulate that energy consumption can be further reduced by maintaining to higher performance power states during computation phases, and lowering CPU frequencies on compute nodes during application I/O phases. We are encouraged by the observation by Ge et al. that energy consumption can be reduced by up to 28% with DVFS for various I/O patterns on a small-scale cluster [15], [18] .

In this paper, we investigate the above mentioned hypothesis on a large-scale supercomputer, where the interconnect network and the I/O subsystem resources are shared by hundreds of concurrent jobs. The compute nodes for running our jobs are not shared by other users. Hence, we reduce the power consumption of CPU cores by setting the power state of a node. We note that other than the compute nodes, all the other resources, i.e., network and I/O subsystem, are unaffected by setting the power state of the nodes. This ensures other jobs running on the system are unaffected. We discuss the details of the system, the I/O kernels we used in this study, and I/O time and power/energy measurement methods in the following section.

IV. Experimental Setup

A. Platform

We performed our experiments using 'Edison', a Cray XC30 located at NERSC. Edison's compute partition consists of 5576 compute nodes. Within each compute node are two 2.4 GHz 12-core Intel Ivy-Bridge processors and 64 GB of 1866-DDR3 DRAM.[1] Compute nodes communicate through a a Cray Aries interconnect [12], which supports injection rates of 10 GB/s bi-directional bandwidth per node. The first two tiers of the Aries' dragonfly topology have ample bandwidth to support the full injection rate. The rank-3 network provides 11 GB/s global bandwidth.

Edison has a total of 7.4 PB of "scratch" storage provided by a Cray Sonexion 1600 Lustre appliance. The aggregate scratch space is partitioned among three scratch file systems. Our experiments are performed on the "scratch3" file system. Scratch3 has 36 object store servers (OSSs), 144 object store targets (OSTs) and provides 3.2 PB capacity with 72 GB/s I/O bandwidth. For all of our experiments, files are striped across all 144 OSTs.

B. I/O Kernels

We use two parallel I/O kernels in this evaluation: VPIC-IO and VORPAL-IO. These kernels are derived from two applications, Vector Particle-In-Cell (VPIC) [5] and VORPAL [30]. These I/O kernels represent two distinct I/O write motifs with different data sizes.

1) VPIC-IO: VPIC is a highly optimized and scalable particle physics simulation developed by Los Alamos National Lab [5]. VPIC-IO uses H5Part [21] to create a file, write eight 1D array variables and close the file. The H5Part API provides a simple veneer for issuing HDF5 calls corresponding to a time-varying, multi-variate particle data model. We extracted all the H5Part function calls of the VPIC code to form the VPIC-IO kernel. The particle data written in the kernel is random data of float data type. The I/O motif of VPIC-IO is a 1D particle array of a given number of particles and each particle has eight variables. For the weak-scaling tests, the I/O kernel writes 8 million particles per MPI process and the total size of the file increases as the number of MPI processes increases. For strong scaling results, we vary the number of particles per MPI process based on the total file size.

2) VORPAL-IO: This I/O kernel is extracted from a computational plasma physics framework application simulating the dynamics of electromagnetic systems, plasmas, and rarefied as well as dense gases, named VORPAL developed by

[1]Edison's memory frequency was recently upgraded from 1600 MHz.

TechX [30]. This benchmark uses H5Block [21] to write non-uniform chunks of 3D data per MPI process. The kernel takes 3D block dimensions (x, y, and z) and the number of components as input. In weak scaling experiments with this kernel, we used 3D blocks of 100x100x60 with different number of processors and the data is written for 20 time steps.

C. Measurements

1) I/O time measurements: The VPIC and VORPAL-I/O kernels initiate data structures of corresponding simulations with random data and write data to file system. Both kernels use MPI-IO in collective I/O mode, where the H5Part/H5Block uses Lustre optimizations. The H5Part/H5Block in the Lustre optimization mode sets the number of MPI-IO aggregators equal to a multiple of the number of Lustre OSTs. As mentioned earlier, we have used all the 144 OSTs available on the Lustre file system we used and have set the stripe size as 32 MB for all the experiments. We have measured the I/O time by using *gettimeofday()* calls before opening the file and after closing the file. This interval includes the time to open a HDF5 file in write mode, to write metadata of HDF5 datasets, to write the data to file system, and to close the file. As shown in Figure 2, we select the maximum I/O time of all the MPI processes assuming all the processes wait until the I/O phase is finished.

2) Energy and Power measurements: Edison compute nodes include a counter that records the total energy used by the node. The Cray power monitoring utility makes the energy counter available to the user through a memory mapped file.

We have written a small library to collect the energy counter data from all nodes in a job. One library function is used to sample and record the initial time and energy counters when the code enters a region of interest. A second function samples the final values and adds these elapsed time and energy to the node's accumulated counters. At the end of the job, the library aggregates the single-node measurements, and prints the total energy, walltime and average power to standard output.

The energy measurement reported by our library does not include contributions from the interconnect or the file system. These are important components of the total energy of our I/O benchmarks, but these are shared resources and it is not possible to distinguish the file system energy used by our job from others on the system.

The Cray PM tools give users control of the frequency, and (indirectly) power used by CPUs on compute nodes. At run time, users may set the CPU frequency for a single job using the `aprun --p-state` option to select a frequency between 1.2 and 2.4 GHz. For example, to run an MPI job with 2048-cores at a `--p-state` of 1.8 GHz, we use the following command, where `exec` and `args` are an MPI application executable name and the arguments of the application.

```
aprun --p-state=1800000 -n 2048 exec args
```

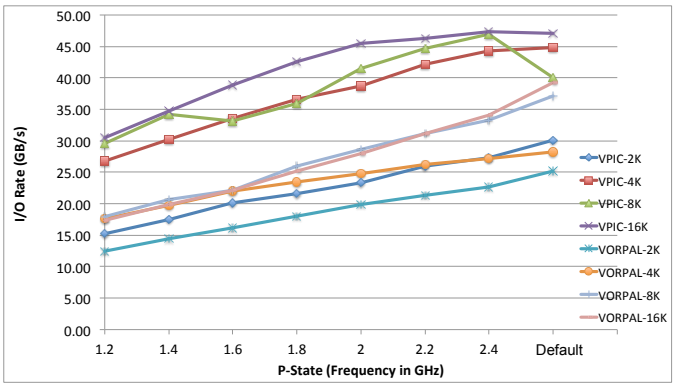

Fig. 3. Weak Scaling of VPIC-IO and VORPAL-IO - I/O Rate in GB/s

V. Results

In this section, we present the I/O rate (in GB/s), power consumption, energy consumption, and energy efficiency (in MB/s/Watt) of VPIC-IO and VORPAL-IO kernels with weak scaling and of VPIC-IO kernel with strong scaling. For weak scaling tests, the data written by each kernel increases proportionately with the number of cores used. We increased the number of cores from 2048 (2K) to 16384 (16K). Since the measured I/O time is the maximum time of all the MPI processes, the I/O rate may be less than the possible I/O bandwidth of the system if an MPI process takes more time to write data than all the other processes. We use the ratio of the I/O rate and the power consumed as the energy efficiency (in MB/s per Watt). In previous studies of I/O energy [15], [18], the same metric was used for measuring energy efficiency. For strong scaling tests, we have studied the VPIC-IO kernel in writing a fixed amount of data at all concurrencies. As the data size does not scale evenly for the VORPAL-IO kernel, we have limited the strong scaling study to the VPIC-IO benchmark. We have varied the frequency of CPUs from 1.2 to 2.4 GHz, with 0.2 GHz increments and compared with the default setting on Edison. The default state dynamically adjusts the CPU frequency between 2.4 and 3.2 GHz, depending on the thermal budget of the chip.

A. Weak Scaling - VPIC I/O and VORPAL I/O

Figures 3, 4, 5, and 6 show the I/O rate, power consumption, energy consumption, and energy efficiency, respectively, of VPIC-IO and VORPAL-IO kernels for different concurrencies with varying CPU frequencies. From Fig. 3, we can see that at a given concurrency, VPIC achieves higher I/O rates than VORPAL-IO, while for a given frequency, VPIC requires fewer cores (4K) to maximize I/O rate than VORPAL-IO (8K). The I/O rates of both kernels roughly double between the 1.2 and 2.4 GHz settings. This is a strong sensitivity and further study is needed to understand the role of CPU activity during the I/O phase.

Figure 4 shows that power consumption increases linearly between 1.2 and 2.4 GHz, and jumps by 25% for the default p-state. Closer inspection shows that the power per node is

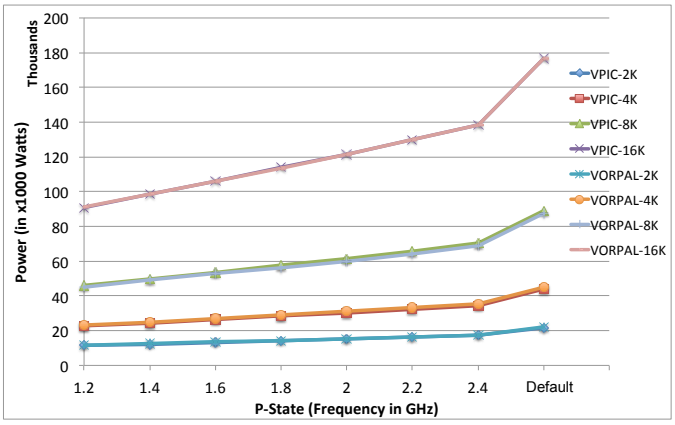

Fig. 4. Weak Scaling of VPIC-IO and VORPAL-IO - Power consumption

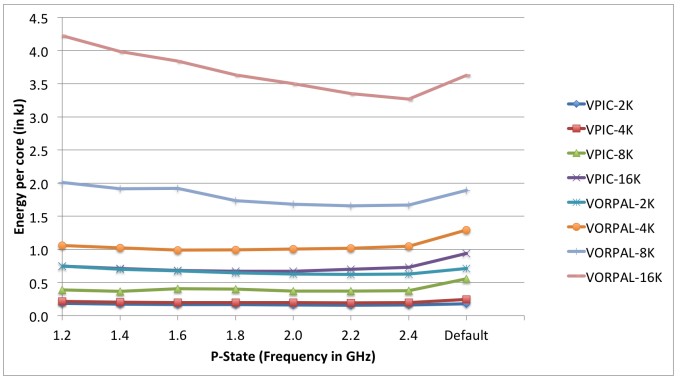

Fig. 5. Weak Scaling of VPIC-IO and VORPAL-IO - Energy consumption per node

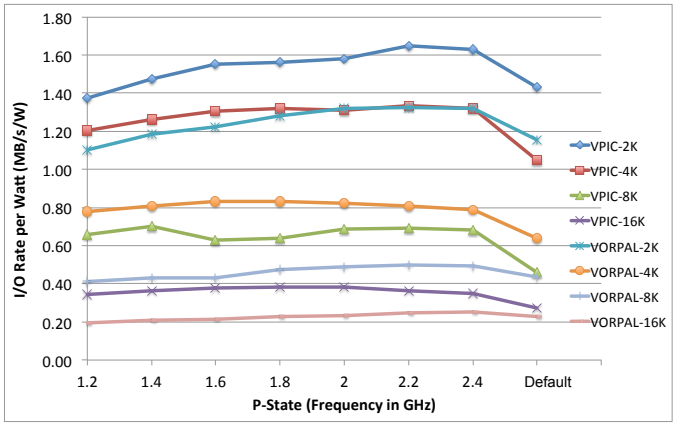

Fig. 6. Weak Scaling of VPIC-IO and VORPAL-IO - Efficiency (I/O rate per Watt [MB/s/Watt])

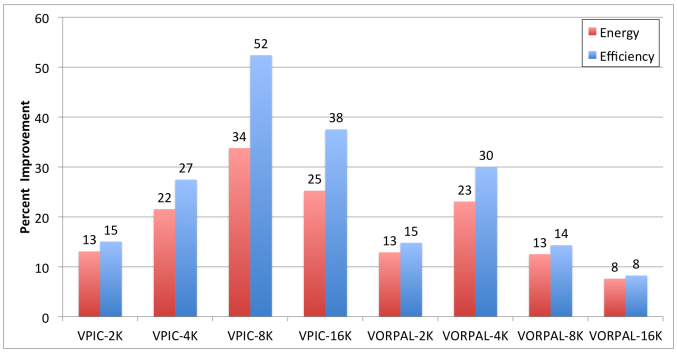

Fig. 7. Percent improvement of energy and energy efficiency for the I/O kernels

independent of concurrency or I/O workload. In Figure 5, the total energy per node for the VPIC I/O benchmark is generally independent of CPU frequency, but the turbo-boost (enabled by the default p-state) increases energy use. Vorpal-IO uses more energy per node at higher concurrencies, and has minimum around 2.4 GHz that become more pronounced when more cores are used.

Fig. 6 shows the energy efficiency (in MB/s/Watt) combining the energy consumption and the I/O rate for the kernels. Similar to energy consumption, the efficiency was highest when the frequency was set at 2.2 GHz for five out of the eight experiments (VPIC-2K, VPIC-4K, VORPAL-2K, VORPAL-8K, VORPAL-16K). The highest energy efficiency for VPIC-8K, and VPIC-16K, VORPAL-4K were observed at 1.4 GHz, at 1.6GHz, and at 1.8GHz, respectively.

In Fig. 7, we show the energy savings for the kernels at different concurrencies compared to the default energy. The energy savings vary between 7.6% and 33% for these kernels. Fig. 7 also shows the energy efficiency improvement (in percent), where we can see that it ranges between 8% and 52%. Further study is needed to reduce the I/O rate degradation with frequency reduction to improve the energy efficiency significantly at lower frequencies.

B. Strong Scaling - VPIC I/O

Figures 8, 9, and 10 show the I/O rate, energy consumption, and energy efficiency of VPIC-IO kernel for a fixed problem size, different concurrencies and variable CPU frequencies. The power per node measurements matched those of Fig. 4.

The I/O rates shown in Fig. 8 increase between 2K and 4K core counts, but not between 4K and 8K. This suggests that performance of the I/O subsystem can be saturated with fewer than 4K cores. However, the I/O rate increases with CPU frequency at all concurrencies tested. This is reflected by the similar energy per core measurements with 8K and 4K cores— both have a broad and shallow (10% of the default p-state) energy minimum around 2.2 GHz. With 2K cores, the total energy use is higher, and the energy minimum is more pronounced (20% of the default p-state) and shifts to 2.4 GHz.

In Fig. 10, we observe the highest energy efficiency at 2.4 GHz for VPIC-2K, at 2.2 GHz for VPIC-4K, but the

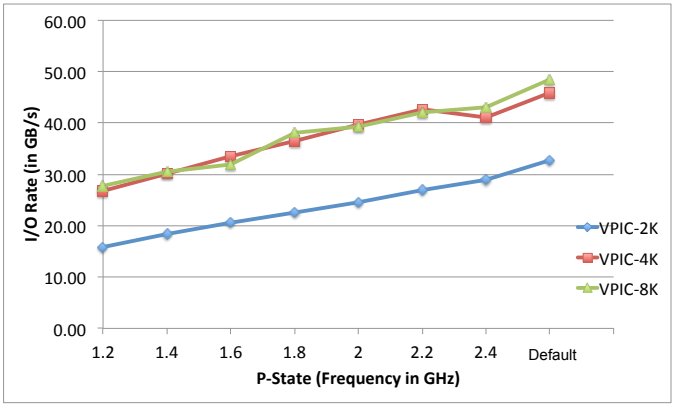

Fig. 8. Strong Scaling of VPIC-IO - I/O Rate in GB/s

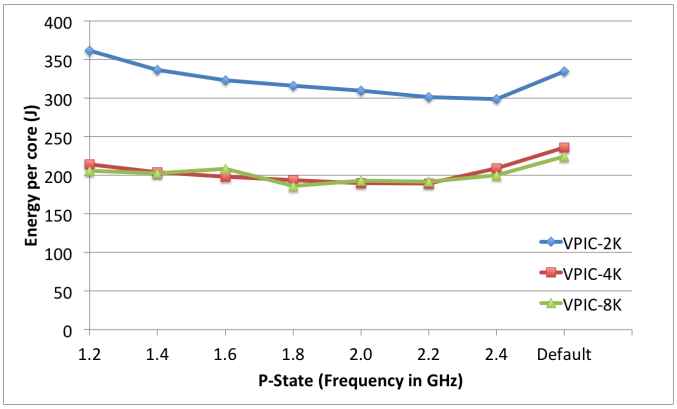

Fig. 9. Strong Scaling of VPIC-IO - Energy consumption per node

energy efficiency of VPIC-8K experiment does not depend strongly on CPU frequency . The energy efficiency improvement is between 12% and 25%. Energy efficiency decreases with concurrency because the I/O rate approaches its maximum between 2K and 4K cores, but power use increases in proportion to node count.

VI. Conclusions

In this study, we presented our initial evaluations of compute node energy consumption with frequency scaling on a Cray XC30 and corresponding parallel I/O performance. We studied the scalability of two I/O kernels, extracted from plasma physics and accelerator physics simulations. We have shown that performance degrades significantly as frequency reduces. However, the energy consumption is minimal neither at the lowest power-state or at the highest. Among our observations with weak scaling of the two I/O kernels, each at three concurrencies, the lowest energy was consumed when we used a CPU frequency of 2.2 GHz compared to the default frequency. While the achieved I/O rate was highest with the default CPU frequency for all the observations, the best energy efficiencies, i.e., I/O rate per Watt, was achieved with 2.2 GHz in 5 out of the 8 observations. The lowest energy and energy efficiency with the other observations was achieved with 1.8 GHz frequency, although those with 2.2 GHz was

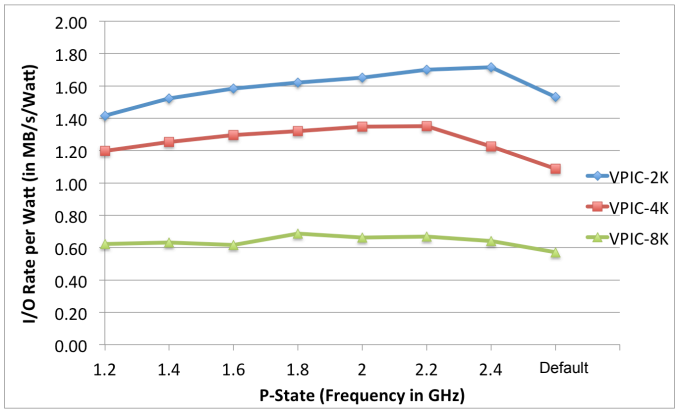

Fig. 10. Strong Scaling of VPIC-IO - Efficiency (I/O rate per Watt [MB/s/Watt])

approximately similar. The energy savings with the best energy efficiency is between 7.6% and 33% and the energy efficiency benefit is between 8.2% and 52%.

While this is the first study of energy and parallel I/O energy efficiency on massive-scale supercomputers, we observed significant variance in I/O rates and energy consumptions. As the I/O subsystem is shared by numerous users, the variation will be notable. Dynamic frequency scaling to accommodate to those variations has potential to save energy further. Understanding these behaviors and applying dynamic scaling requires further investigation. Another area to investigate include setting the power states only for the cores, instead of setting them for an entire compute node. We suspect that memory controller and DRAM also are moving to lower power state degrade I/O performance substantially. We will investigate the impact of fine grain frequency scaling to avoid I/O performance degradation. We will also explore MPI-IO settings, such as independent I/O mode and setting the aggregators in collective I/O based on topology.

Acknowledgment

This work is supported by the Director, Office of Science, Office of Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. This research used resources of the National Energy Research Scientific Computing Center.

References

[1] N. Ali, P. Carns, K. Iskra, D. Kimpe, S. Lang, R. Latham, R. Ross, L. Ward, and P. Sadayappan. Scalable I/O forwarding framework for high-performance computing systems. In *Cluster Computing and Workshops, 2009. CLUSTER '09. IEEE International Conference on*, pages 1–10, Aug 2009.

[2] B. Austin and N. J. Wright. Measurement and interpretation of microbenchmark and application energy use on the cray XC30. In *Proceedings of the 2nd International Workshop on Energy Efficient Supercomputing*, E2SC '14, pages 51–59, 2014.

[3] B. Behzad, H. V. T. Luu, et al. Taming Parallel I/O Complexity with Auto-tuning. In *Proceedings of SC13: International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '13, pages 68:1–68:12, New York, NY, USA, 2013. ACM.

[4] K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Karp, S. Keckler, D. Klein, R. Lucas, M. Richards, A. Scarpelli, S. Scott, A. Snavely, T. Sterling, R. S. Williams, K. Yelick, K. Bergman, S. Borkar, D. Campbell, W. Carlson, W. Dally, M. Denneau, P. Franzon, W. Harrod, J. Hiller, S. Keckler, D. Klein, P. Kogge, R. S. Williams, and K. Yelick. ExaScale Computing Study: Technology Challenges in Achieving Exascale Systems Peter Kogge, Editor and Study Lead. http://www.cse.nd.edu/Reports/2008/TR-2008-13.pdf, 2008.

[5] K. J. Bowers, B. J. Albright, L. Yin, B. Bergen, and T. J. T. Kwan. Ultrahigh performance three-dimensional electromagnetic relativistic kinetic plasma simulation. *Physics of Plasmas*, 15(5):7, 2008.

[6] S. Breitenfeld, K. Chadalavada, R. Sisneros, S. Byna, Q. Koziol, N. Fortner, Prabhat, and V. Vishwanath. Recent Progress in Tuning Performance of Large-scale I/O with Parallel HDF5. In *Proceedings of the 9th Parallel Data Storage Workshop*, PDSW '14, 2014.

[7] H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, et al. VisIt: An End-User Tool for Visualizing and Analyzing Very Large Data. 2013.

[8] J. Chou, J. Kim, and D. Rotem. Energy-aware scheduling in disk storage systems. In *2011 International Conference on Distributed Computing Systems, ICDCS 2011, Minneapolis, Minnesota, USA, June 20-24, 2011*, pages 423–433, 2011.

[9] J. Chou, T.-H. Lai, J. Kim, and D. Rotem. Exploiting replication for energy-aware scheduling in disk storage systems. *Parallel and Distributed Systems, IEEE Transactions on*, PP(99):1–1, 2014.

[10] Getting Started with MPI I/O. http://docs.cray.com/books/S-2490-40/S-2490-40.pdf.

[11] B. Dong, S. Byna, and K. Wu. SDS: A Framework for Scientific Data Services. In *Proceedings of the 8th Parallel Data Storage Workshop*, PDSW '13, pages 27–32, New York, NY, USA, 2013. ACM.

[12] G. Faanes, A. Bataineh, D. Roweth, T. Court, E. Froese, B. Alverson, T. Johnson, J. Kopnick, M. Higgins, and J. Reinhard. Cray Cascade: A Scalable HPC System Based on a Dragonfly Network. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, SC '12, pages 103:1–103:9, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.

[13] G. Fourestey, B. Cumming, L. Gilly, and T. C. Schulthess. First Experiences With Validating and Using the Cray Power Management Database Tool. *CoRR*, abs/1408.2657, 2014.

[14] V. W. Freeh, F. Pan, N. Kappiah, D. Lowenthal, and R. Springer. Exploring the Energy-Time Tradeoff in MPI Programs on a Power-Scalable Cluster. In *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, pages 4a–4a, April 2005.

[15] R. Ge. Evaluating parallel I/O energy efficiency. In *Green Computing and Communications (GreenCom), 2010 IEEE/ACM Int'l Conference on Int'l Conference on Cyber, Physical and Social Computing (CPSCom)*, pages 213–220, Dec 2010.

[16] R. Ge, X. Feng, and K. Cameron. Performance-constrained Distributed DVS Scheduling for Scientific Applications on Power-aware Clusters. In *Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference*, Nov 2005.

[17] R. Ge, X. Feng, W. chun Feng, and K. Cameron. CPU MISER: A Performance-Directed, Run-Time System for Power-Aware Clusters. In *Parallel Processing, 2007. ICPP 2007. International Conference on*, Sept 2007.

[18] R. Ge, X. Feng, and X.-H. Sun. SERA-IO: Integrating Energy Consciousness into Parallel I/O Middleware. In *Cluster, Cloud and Grid Computing (CCGrid), 2012 12th IEEE/ACM International Symposium on*, pages 204–211, May 2012.

[19] F. Gruian and K. Kuchcinski. LEneS: task scheduling for low-energy systems using variable supply voltage processors. In *Design Automation Conference, 2001. Proceedings of the ASP-DAC 2001. Asia and South Pacific*, pages 449–455, 2001.

[20] S. Herbert, S. Garg, and D. Marculescu. Exploiting process variability in voltage/frequency control. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 20(8):1392–1404, Aug 2012.

[21] M. Howison, A. Adelmann, E. W. Bethel, A. Gsell, B. Oswald, and Prabhat. H5hut: A High-Performance I/O Library for Particle-Based Simulations. In *Proceedings of 2010 Workshop on Interfaces and Abstractions for Scientific Data Storage (IASDS10)*, Heraklion, Crete, Greece, Sept. 2010. LBNL-4021E.

[22] C.-H. Hsu and W. chun Feng. A feasibility analysis of power awareness in commodity-based high-performance clusters. In *Cluster Computing, 2005. IEEE International*, pages 1–10, Sept 2005.

[23] C.-H. Hsu and W. chun Feng. A power-aware run-time system for high-performance computing. In *Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference*, Nov 2005.

[24] J. Kim, J. Chou, and D. Rotem. iPACS: Power-aware covering sets for energy proportionality and performance in data parallel computing clusters. *Journal of Parallel and Distributed Computing*, 74(1):1762–1774, 2014.

[25] Kitware. ParaView. http://www.paraview.org/.

[26] B. Li, S. Song, I. Bezakova, and K. Cameron. EDR: An energy-aware runtime load distribution system for data-intensive applications in the cloud. In *Cluster Computing (CLUSTER), 2013 IEEE International Conference on*, pages 1–8, Sept 2013.

[27] J. Li and J. Martinez. Dynamic power-performance adaptation of parallel computation on chip multiprocessors. In *High-Performance Computer Architecture, 2006. The Twelfth International Symposium on*, pages 77–87, Feb 2006.

[28] A. Manzak and C. Chakrabarti. Variable voltage task scheduling algorithms for minimizing energy. In *Low Power Electronics and Design, International Symposium on, 2001.*, pages 279–282, 2001.

[29] S. Martin and M. Kappel. Cray XC30 Power Monitoring and Management. In *Cray User Group meeting proceedings*, 2014.

[30] C. Nieter and J. R. Cary. VORPAL: a versatile plasma simulation code. *Journal of Computational Physics*, 196:448–472, 2004.

[31] S. W. Skillman, M. S. Warren, M. J. Turk, R. H. Wechsler, D. E. Holz, and P. M. Sutter. Dark Sky Simulations: Early Data Release. *ArXiv e-prints*, July 2014.

[32] S. Son, M. Kandemir, and A. Choudhary. Software-directed disk power management for scientific applications. In *Parallel and Distributed Processing Symposium, 2005. Proceedings. 19th IEEE International*, April 2005.

[33] S. Song, C.-Y. Su, R. Ge, A. Vishnu, and K. Cameron. Iso-Energy-Efficiency: An Approach to Power-Constrained Parallel Computation. In *Parallel Distributed Processing Symposium (IPDPS), 2011 IEEE International*, pages 128–139, May 2011.

[34] The HDF Group. HDF5 user guide. http://hdf.ncsa.uiuc.edu/HDF5/doc/H5.user.html, 2010.

[35] Unidata. The NetCDF users' guide. http://www.unidata.ucar.edu/software/netcdf/docs/netcdf/, 2010.

[36] L. Wang, S. U. Khan, D. Chen, J. KoOdziej, R. Ranjan, C.-Z. Xu, and A. Zomaya. Energy-aware parallel task scheduling in a cluster. *Future Generation Computer Systems*, 29(7):1661–1670, Sept. 2013.

[37] G.-Y. Wei, J. Kim, D. Liu, S. Sidiropoulos, and M. Horowitz. A variable-frequency parallel I/O interface with adaptive power-supply regulation. *Solid-State Circuits, IEEE Journal of*, 35(11):1600–1610, Nov 2000.

[38] F. Yao, A. Demers, and S. Shenker. A scheduling model for reduced CPU energy. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 374–382, Oct 1995.