Using Multitasking and SSD Disks for Optimising Computing Cluster Energy-Efficiency

Tapio Niemi
Helsinki Institute of Physics, Technology Programme
CERN, CH-1211 Geneva 23, Switzerland
tapio.niemi@cern.ch

Jukka Kommeri
Helsinki Institute of Physics, Technology Programme
CERN, CH-1211 Geneva 23, Switzerland
kommeri@cern.ch

Ari-Pekka Hameri
HEC, University of Lausanne, CH-1015 Lausanne, Switzerland
ari-pekka.hameri@unil.ch

***Abstract*—We tested how solid-state drives (SSD) and multitasking, i.e., running more than one task per CPU core, affect performance and energy consumption in I/O intensive data analysis jobs in high-energy physics (HEP) computing. Our motivation for the study comes from the LHC experiment at CERN producing up to 15 petabytes of data annually. The data is analysed by a huge grid computing infrastructure containing more than 100 000 CPU cores over 140 computing centres in different countries.**

Our tests indicated that in I/O intensive HEP computing multitasking, mixing heterogeneous tasks, and using SSD disks can clearly improve throughput and lower energy consumption. The throughput improved around 120% and energy consumption decreased around 50% compared to a conventional one task per CPU core setting with a hard disk system.

***Keywords*-energy-efficiency; multitasking; solid-state disk (SDD); scientific computing.**

I. INTRODUCTION

Energy consumption has become one of the main costs of computing and several methods to improve the situation has been suggested. The focus of the research in this field has been in hardware and infrastructure aspects, probably because of most of the computing centres focus on high-performance computing trying to optimise processing time of individual computing jobs. Jobs can have strict deadlines or require massive parallelism. Instead, in so called high-throughput computing the aim is slightly different, since individual tasks are not time critical and the aim is to optimise the total throughput over a longer period of time. This opens new possibilities for system optimisation, too. In the current work, our focus is closer to high-throughput computing than high-performance computing.

In modern multi-core servers, I/O access easily becomes a bottleneck. While prices of solid-state drives (SSD) are coming down and their capacity becoming larger, they can be used to alleviate this problem. SSD disks clearly have several benefits over conventional hard disks but it is not clear how much better they are in different application areas [1]. In this paper, we study applicability of SSD disks in data analysis needed in high-energy physics (HEP). Our focus is on a typical grid computing problem: How to optimise processing a large set of data intensive tasks for both energy consumption and throughput. By energy consumption we mean how much electricity is needed to process a task or a job, while throughput means how many similar tasks can be processed in a time unit. A computer or method is more energy-efficient than another one if it uses less energy for processing the same set of similar jobs.

In data and computing intensive sciences, such as high energy-physics, optimised and energy-efficient solutions are important. For example, the Worldwide LHC Computing Grid (WLCG[1]) being used for analysing the data that the Large Hadron Collier of CERN will produce, includes hundreds of terabytes of storage and tens of thousands CPU cores. In this scale, even a small system optimisation can offer noticeably energy and cost savings and performance improvements. Since high-energy physics computing has many special characteristics, common industry practices are not always the best but the methods must be tested and optimised especially for HEP computing. The main characteristics in this sense are: jobs contain large sets of similar tasks, data-intensive computing, processing times of individual tasks are not important but the aim is to optimise the total processing time of a set of hundreds or thousands of jobs each containing up to thousands of tasks, no preceding conditions among tasks, and little intercommunication between jobs or their tasks, i.e., high parallelisms. In spite of this special nature, optimising HEP computing and clusters has mostly focused on similar infrastructure issues as HPC computing in general such as cooling and purchasing energy-efficient hardware.

The paper has been organised as follows. After introduction, we review the related work in Section II. Our methodology is described in Section III. Then, the test environment and tests are explained in Section IV and their results given in Section V. Finally, conclusions and future work are given in Section VI.

[1] http://lcg.web.cern.ch/lcg/public

 ISBN: 978-1-61208-136-6

II. Related work

Generally, we can say that optimisation of scientific computing facilities has mostly focused on hardware and infrastructure issues, not so much on operational methods such as workload management and even less on operating system or application software optimisation. These methods mostly try to decrease energy consumption and keep computing power constant, while our aim is to increase utilisation rate of CPUs and I/O and in this way increase throughput, which decreases average power consumed by a computing job.

Scheduling is a widely studied topic but most of the work focuses on finding optimal schedules when jobs have preceding constrains and/or strict time limitations. Optimising total throughput or energy efficiency in high throughput computing has received less research interest. Instead some works suggest clearly opposite approaches: For example, Koole and Righter [2] suggest a scheduling model in which tasks are replicated to several computers.

Goes et al. [3] have studied scheduling of irregular I/O intensive parallel jobs. They note that CPU load alone is not enough but all other system resources (memory, network, storage) must be taken into account in scheduling decisions. Wang et al. [4] have studied optimal scheduling methods in a case of identical jobs and different computers. They aimed to maximise the throughput and minimise the total load. They give an on-line algorithm to solve the problem. There are also some studies on energy-aware scheduling. For example Bunde [5] has studied power aware scheduling methods for minimising energy consumption and not reducing system performance by applying dynamic voltage scaling technologies. However, none of these works directly aims at maximising throughput and minimising energy consumption.

Another group of studies focuses on the server level. These studies have more technology oriented approach than ours. A detailed overview on techniques that can be used to reduce energy consumption of computer systems was given by Venkatachalam and Franz [6]. Other work in this group includes, for example, Li et al. [7], who studied performance guaranteed control algorithms for energy management of disk and main memory. Ge et al. [8] studied methods based on Dynamic Voltage Scaling technology of microprocessors and created a software framework to implement and evaluate their method. Finally, Essary and Amer [9] have studied how disk arm movements can be optimised and in this way save energy, while Zhu et al. [10] proposed a disk array energy management system.

There is some recent research directly related to SSD disks. Rizvi and Chung have studied an application of solid-state drives (SSD) in large databases [11]. They showed that SSDs can improve both read and write throughput of the database mostly because of low latency of SSD disk in random data access. Schmidt et al. [12] studied XML database systems with SSD disks. In addition to performance, they also focused on energy-efficiency. Narayanan et al. [13] have analysed whether SSD disks are beneficial for server usage. They developed a tool for analysing storage workload and determining the best storage solution. They found out that SSD disks are much more energy-efficient than conventional hard disks but their current price does not make them a cost-efficient solution yet. Kim et al. [14] studied different disk scheduling methods for SSD disk. They noticed that common Linux disk schedulers are optimised more for hard disks than SSD disks. To improve the situating they proposed two new schedulers optimised for SSD disks. Chen et al. [1] studied performance of SSD disks. Their tests confirmed many commonly understood properties of SSDs but they also noticed many challenges in their performance. Generally, there is little work studying SSD disks in high-throughput computing, although modern multi-core servers obviously could benefit from faster disk access. In our current work, we aim at showing that this hypothesis is valid.

III. Methodology

Generally, HEP computing consists of two kinds of computing jobs: 1) simulation jobs modelling collisions in particle colliders and 2) analysis of data collected by detectors. In LHC computing there are two kinds of sources of these jobs: 1) centralised production teams that send up to millions of similar long simulation jobs, and 2) thousands of individual scientists sending analysis jobs in a non-controlled way. In analysis jobs data storage and transfer becomes a challenge. The data storage of the LHC experiments follows a so called tier model, in which the data is replicated from tier-0 via tier-1 and tier-2 centres to final users. Because of relatively slow data access over the Internet, the data is stored locally before analysis. Often it is beneficial to first copy a relevant piece of data to the local disk of the computing node, since the cluster file system is easily slower in a heavy use than the local disk. This is beneficial especially if the same data is iteratively analysed by only changing some parameters. Our study is the most relevant in this kind of use case.

Our earlier work showed that multitasking improves efficiency in HEP data analysis [15] but it also easily makes disk access a bottleneck. Since SSD disks are clearly faster than hard disk, they can be used to partially remove this bottleneck. In the current study, our aim was to test whether SSD disks can be used for improving throughput lowering energy consumption in HEP data analysis. As explained above, we assume having a large set of independent tasks, and we are interested in the total processing time of this set. By the total processing time we mean the total time from the submission of the first task of the set to the computing cluster to the end of processing of the last task of the same set. Our aim is to minimise both the amount of electricity and time needed to process the set. Minimising the processing time is equal to maximising the throughput.

We can present our research problem in a form of the following three hypotheses: Throughput can be improved and electricity consumption reduced in data intensive HEP computing compared to the common single task per CPU core processing with conventional hard disks by:

 ISBN: 978-1-61208-136-6

1) multitasking, i.e., processing more than one task per CPU core in parallel,
2) using SSD disks, and
3) mixing heterogeneous tasks while multitasking.

IV. TESTS

The test applications were real HEP analysis and simulation applications based on the CMSSW framework [16]. We used simulation jobs that ranged from CPU intensive to memory intensive. Our test method was to execute a job, i.e., a large set of tasks and measure the time and electricity consumed during the test run. The tests were run with different disks and configurations of the workload management system.

Our experiment system comprised of one front-end computer running the workload management system, one computing node (a Dell PowerEdge R410 server), 1 Gb local area network, and an electricity meter. The computing node was equipped with two Intel Xeon E5520 quad core 2.27 GHz processors and 16 GB DDR3 memory. The same hardware was used with both a normal magnetic disk and a solid state drive. In our tests only one drive was connected at a time. As a normal magnetic disk we used Seagate Barracuda ES.2 that came with the server. As a solid state drive we had an Intel's X25-M second generation 80GB drive. Both the hard disk and SSD disk were connected through the on-board SATA second generation connector. We used Rocks 5.3 with Linux Kernel 2.6.18 and Ext3 file system. Both disks used the CFQ (Completely Fair Queuing) scheduler, the default I/O scheduler of the Linux kernel. The electricity consumption of the computing nodes was measured with the Watts Up Pro electricity meter. The accuracy of the meter is around +-1%. We used Sun Grid Engine (SGE) workload management system [17] of Sun Microsystems that is also commonly used in grid computing clusters.

Our test applications were:

- *CPU intensive* jobs ran CMSSW [16] including event generation with Pythia6 [18] and full detector simulation with Geant4 [19]. The Pythia program is a standard tool for the generation of high-energy collisions using monte carlo methods. We had two versions of CPU intensive jobs: long and medium. The only difference between them is the run time.
- *Memory intensive* jobs are similar to CPU intensive ones but their memory requirement has been made larger by modifying the configuration file.
- *Data analysis* job is a data analysis application of the CMSSW framework [20]. Input data for the test was from the CRAFT (CMS Running At Four Tesla) experiment that used cosmic ray data recorded with CMS detector at the LHC during 2008 [21]. This detector was used in the similar way to current LHC experiments and the data was very close to the final data analysis. The analysis software reads the input file (94 MB or 360 MB), performs the analysis, and writes a small summary file (<10kB) on the local disk. The I/O traffic of one analysis job is shown in Figure 1.

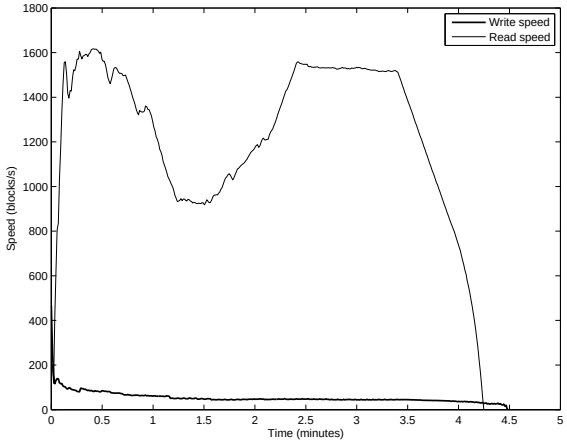

Fig. 1. I/O usage of a HEP analysis job

A summary of test application can be seen in Table I. The output files of all applications are very small, less than 10 kB.

TABLE I
SUMMARY OF HEP TEST APPLICATIONS

Program	Input file	Max mem	Run time
MEM intensive	-	859MB	449s
CPU medium	-	532MB	463s
CPU long	-	548MB	757s
data analysis heavy	360MB	368MB	99s
data analysis light	94MB	271MB	46s

The test applications were combined into three different test sets containing from 3 to 7 applications: 1) I/O intensive set (50 $\times$ 3 data analysis applications), 2) CPU intensive set (50 $\times$ 2 memory and 2 CPU intensive applications), and 3) Mixed set (50 $\times$ 3 data analysis, 2 memory, and 2 CPU applications). Each of these test sets were run 50 times to get results stable enough. Then these sets of 50 runs were run 3 times each and the final results were averages of these runs. The input data was different for each run to eliminate the effect of the disk cache. Additionally the disk cache was cleared between the test sets to provide a similar environment for all test sets. Since differences in averages among the different test sets were relatively high and standard deviation inside the sets low, this test setting gave us large enough sample size to look for statistically significant results.

V. RESULTS

First, we tested the effect of multitasking with different numbers of parallel homogeneous tasks in a core. Figure 2 shows how throughput and energy consumption develops when the number of tasks increases in a computing node. This result clearly shows that Hypothesis 1 is valid. The reason for improvements is not straightforward, since intuitively the result could be partially opposite: if disk access is the bottleneck, overloading it should slow down processing times. However, it

 ISBN: 978-1-61208-136-6

seems to be that this overloading improves disk performance. We assume two possible reasons: 1) Overloading keeps the bottleneck busy and eliminates the total I/O waiting time in the system, and/or 2) overloading improves the efficiency of disk caches since it is more likely that some other task has already fetched the required data.

Figure 3 shows the effect of multitasking with HDD and SSD disks. The figure clearly indicates that the best performance can be achieved using SSD and multitasking together.

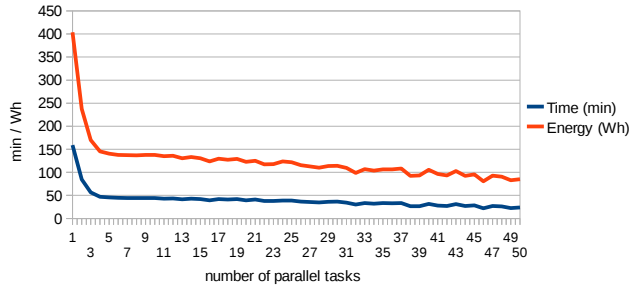

Fig. 2. Processing time of I/O jobs with different amount of parallelism using SSD

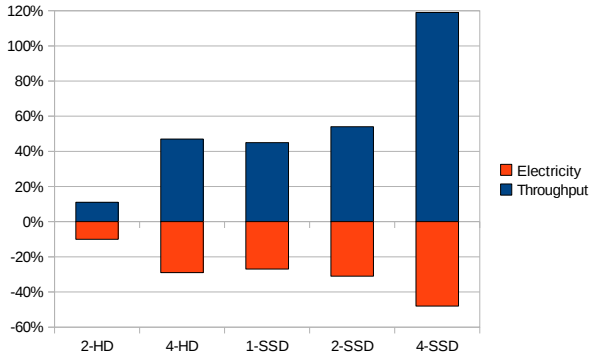

Fig. 3. Improvements of multitasking and SSD compared to 1 task/core with a hard disk

Next we compared the difference between an SSD and a hard disk with different HEP applications. The used scheduling method was two jobs per CPU core. Our earlier research [22], [15] indicates this setting is good in practical cases. It does not usually use too much memory and it is still clearly more efficient than the default scheduling settings of SGE, i.e., job slots being equal to the number of CPU cores. Figure 4 shows run times and electricity consumption of our three test sets in cases of an SSD disk and a hard disk. The numbers are averages of three independent runs. We can see that the differences between the SSD and the hard disk were clear in the I/O test set and smaller in other sets. Improvements of using an SSD disk compared to a conventional hard disk are shown in Table II and illustrated in Figure 5.

In all cases using SSD disks decreased electricity consumption per job but improvements heavily depended on the use-case. With I/O intensive jobs, throughput was 38% better and the energy consumption per job was 25% lower when using the SSD disk instead of the hard disk. The reason for this is much higher read speed and shorter access time of the SSD disk. The higher throughput caused higher CPU utilisation level and the computer using more power (average 174 W instead of 166 W with HD) but much shorter processing time lowered the total electricity consumption by 25%. Generally, the CPU power consumption dominates the disk consumption. In the case of the mixed workload, improvements both in energy efficiency and throughput were small, around 1-2%, but still statistically significant even in our small sample. In the case of pure CPU intensive workload using the SSD disk decreased energy consumption by 2% compared to the hard disk, also throughput was slightly improved but the improvement was not statistically significant.

The test results validate Hypothesis 2 by indicating that in I/O intensive HEP computing an SSD disk clearly outperformed a hard disk both in energy-efficiency and throughput. With the mixed workload the difference was less significant, since mixing I/O and CPU intensive tasks makes the disk speed less important. However, the SSD disk never decreased performance.

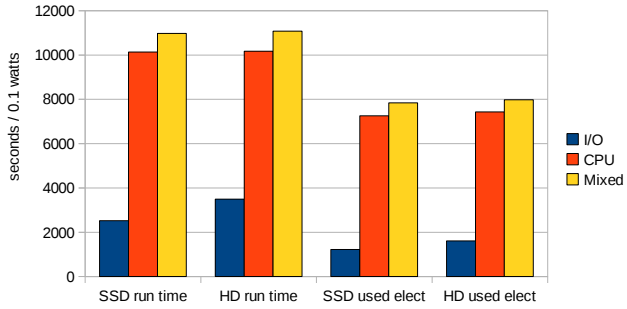

Fig. 4. Processing times and electricity consumption of different test sets

Finally, we tested the effect of mixing I/O and CPU intensive jobs when using the same disk. The second last rows of both disks in Table II show the results when the jobs were mixed while the last rows show the combined numbers of both I/O and CPU intensive job sets. This corresponds to a case in which all I/O jobs are scheduled to be processed before all CPU jobs. The results indicated that mixing jobs using different resources clearly improved both throughput and energy-efficiency. In this test case the hard disk gave better improvements but the SSD disk was still absolutely better. The results validate Hypothesis 3.

VI. Conclusions and Future Work

Our tests showed that throughput can be improved and energy consumption reduced using multitasking together with an SSD disk. However, the results depended highly on the used application. The best performance was achieved by combining

 ISBN: 978-1-61208-136-6

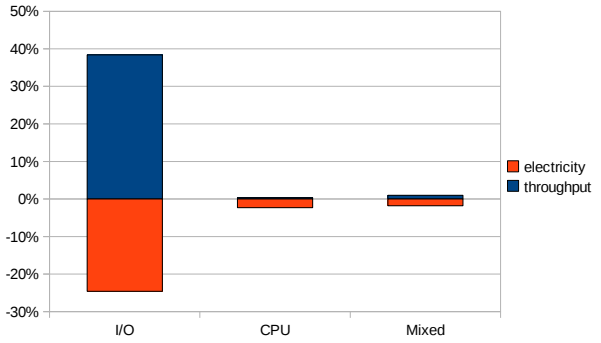

Fig. 5. Improvements of SSD compared to conventional HD (2 task/core setting)

TABLE II
COMPARISON BETWEEN HD AND SSD (2 TASK/CORE) SETTING

Disk	Test	jobs/h	%	Wh/job	%	Avg W
HD	I/O	51.6		3.2		166
HD	CPU	17.7		14.9		263
HD	Mixed	16.3		16.0		259
HD	I/O+CPU	13.2		18.1		238
SSD	I/O	71.4	38	2.4	-25	174
SSD	CPU	17.8	0.34	14.5	-2.3	258
SSD	Mixed	16.4	0.98	15.7	-1.8	257
SSD	I/O+CPU	14.2	7.8	17.0	-6.3	241

multitasking and mixing different workloads such as I/O intensive analysis and CPU intensive simulation. This improved the total throughput and decreased the energy consumption in both disk solutions.

Our tests showed that multitasking with SSDs can clearly lower energy consumption and increase throughput in I/O intensive computing compared to the hard disk. In the case of the mixed or CPU intensive workload the difference was less significant. When processing two tasks per CPU core and using an SSD disk, in I/O intensive physics data analysis throughput was 38% higher and energy consumption 25% lower compared to a hard disk, while in CPU intensive computing the disk solution makes only a couple of percentages difference. When increasing multitasking, the results got even better, but then the risk of over using memory becomes higher. The aim of our study was not to explain why multitasking improves performance but we can assume that the reason is better total utilisation of resources: the disk access is an obvious bottleneck in data analysis and processing several tasks in parallel keeps the bottleneck busy, since all the time some task is performing a disk operation. Another reason may be a better utilisation of the disk cache. Multitasking, of course, increases the processing time of an individual task, since there is often a queue for the disk access but the total processing time of a set of tasks, i.e., a job, becomes shorter.

Today SSD disks are still significantly more expensive per gigabyte than hard disks (the cheapest SSDs around $2.5/GB, HDs $0.15/GB), and therefore they may not improve total cost-efficiency if lots of storage space is needed. In some cases, a small SSD disk may be enough in a computing node and thus it can be a cost-efficient solution if using it removes an I/O bottleneck and in this way dramatically improves utilisation rates of other components. In 2010, a small (80GB) SSD disk costs around $250 and a small hard disk (160GB) around $50. This difference is only around 10% of the total price of the server but, according to our study, it can give up to 38% more performance in data intensive computing. This means reducing energy cost by 25% per computing job and makes it also possible to save in hardware, infrastructure, and personnel costs. Further, SSD disk storage is still around 10 times cheaper than RAM memory and in many applications a non permanent RAM disk is not a working solution. Although SSD market is fairly new, the price per performance keeps improving and the role of the price of SSD in total cost of the computing node becomes less important.

Our future work will include studying how different disks, disk schedulers, and task scheduling methods affect performance and energy-efficiency of HEP computing applications. We are also developing a simulator to determine an optimum amount of multitasking for different computing tasks and methods profiling analysis jobs.

ACKNOWLEDGEMENTS

We would like to thank Magnus Ehrnrooth's Foundation for a grant for the test hardware and Matti Kortelainen of the Helsinki Institute of Physics for helping with physics applications.

REFERENCES

[1] F. Chen, D. A. Koufaty, and X. Zhang, "Understanding intrinsic characteristics and system implications of flash memory based solid state drives," in *SIGMETRICS '09: Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems*. New York, NY, USA: ACM, 2009, pp. 181–192.

[2] G. Koole and R. Righter, "Resource allocation in grid computing," *J. Scheduling*, vol. 11, no. 3, pp. 163–173, 2008.

[3] L. F. Ges and et al., "Anthillsched: A scheduling strategy for irregular and iterative i/o-intensive parallel jobs," in *Job Scheduling Strategies for Parallel Processing - JSSPP 2005*. Springer, 2005.

[4] C.-M. Wang, X.-W. Huang, and C.-C. Hsu, "Bi-objective optimization: An online algorithm for job assignment," in *GPC 2009, Geneva, Switzerland*, 2009, pp. 223–234.

[5] D. P. Bunde, "Power-aware scheduling for makespan and flow," in *SPAA '06: Proceedings of the 18th annual ACM symposium on Parallelism in algorithms and architectures*. New York, NY, USA: ACM, 2006, pp. 190–196.

[6] V. Venkatachalam and M. Franz, "Power reduction techniques for microprocessor systems," *ACM Comput. Surv.*, vol. 37, no. 3, pp. 195–237, 2005.

[7] X. Li, Z. Li, Y. Zhou, and S. Adve, "Performance directed energy management for main memory and disks," *Trans. Storage*, vol. 1, no. 3, pp. 346–380, 2005.

[8] R. Ge, X. Feng, and K. W. Cameron, "Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters," in *SC '05: Proceedings of the 2005 ACM/IEEE conference on Supercomputing*. Washington, DC, USA: IEEE Computer Society, 2005, p. 34.

[9] D. Essary and A. Amer, "Predictive data grouping: Defining the bounds of energy and latency reduction through predictive data grouping and replication," *Trans. Storage*, vol. 4, no. 1, pp. 1–23, 2008.

 ISBN: 978-1-61208-136-6

[10] Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes, "Hibernator: helping disk arrays sleep through the winter," in *SOSP '05, 20th ACM symposium on Operating systems principles*. New York, NY, USA: ACM, 2005, pp. 177–190.

[11] S. S. Rizvi and T.-S. Chung, "Flash memory ssd based data management for data warehouses and data marts," *Convergence Information Technology, International Conference on*, vol. 0, pp. 858–860, 2009.

[12] K. Schmidt, Y. Ou, and T. Härder, "The promise of solid state disks: increasing efficiency and reducing cost of dbms processing," in *C3S2E '09: Proceedings of the 2nd Canadian Conference on Computer Science and Software Engineering*. New York, NY, USA: ACM, 2009, pp. 35–41.

[13] D. Narayanan, E. Thereska, A. Donnelly, S. Elnikety, and A. Rowstron, "Migrating server storage to ssds: analysis of tradeoffs," in *EuroSys '09: Proceedings of the 4th ACM European conference on Computer systems*. New York, NY, USA: ACM, 2009, pp. 145–158.

[14] J. Kim, Y. Oh, E. Kim, J. Choi, D. Lee, and S. H. Noh, "Disk schedulers for solid state drivers," in *EMSOFT '09: Proceedings of the seventh ACM international conference on Embedded software*. New York, NY, USA: ACM, 2009, pp. 295–304.

[15] T. Niemi, J. Kommeri, and H. Ari-Pekka, "Energy-efficient scheduling of grid computing clusters," in *Proceedings of the 17th Annual International Conference on Advanced Computing and Communications (ADCOM 2009), Bengaluru, India*, 2009.

[16] F. Fabozzi, C. D. Jones, B. Hegner, and L. Lista, "Physics analysis tools for the CMS experiment at LHC," *IEEE Trans. Nucl. Sci.*, vol. 55, pp. 3539–3543, 2008.

[17] *BEGINNER'S GUIDE TO SUNTM GRID ENGINE 6.2 Installation and Configuration*, Sun Microsystems, 2008.

[18] T. Sjostrand, S. Mrenna, and P. Z. Skands, "PYTHIA 6.4 Physics and Manual," *JHEP*, vol. 05, p. 026, 2006. [Online]. Available: http://www.slac.stanford.edu/spires/find/hep/www?key=6566170{\&}FORMAT=WWWBRIEFBIBTEX

[19] S. Agostinelli *et al.*, "GEANT4: A simulation toolkit," *Nucl. Instrum. Meth.*, vol. A506, pp. 250–303, 2003.

[20] CMS_Experiment, *CMSSW Application Framework*, https://twiki.cern.ch/twiki/bin/view/CMS/WorkBookCMSSWFramework.

[21] D. Acosta and T. Camporesi, "Cosmic success," *CMS Times*, November 2008.

[22] T. Niemi, J. Kommeri, K. Happonen, J. Klem, and A.-P. Hameri, "Improving energy-efficiency of grid computing clusters," in *Advances in Grid and Pervasive Computing, 4th International Conference, GPC 2009, Geneva, Switzerland*, 2009, pp. 110–118.

 ISBN: 978-1-61208-136-6