Energy Consumption Estimation in Parallel Applications: an Analysis in Real and Theoretical Models

Dieison S. Silveira[1], **Gabriel B. Moro**[2], **Eduardo H. M. da Cruz**[2]
Philippe O. A. Navaux[2], **Lucas Mello Schnorr**[2], **and Sergio Bampi**[1,2]

[1]Graduate Program in Microelectronics (PGMicro)
[2] Graduate Program in Computing (PPGC)
Federal University of Rio Grande do Sul (UFRGS) – Porto Alegre – RS – Brazil

{dssilveira, gbmoro, ehmcruz, navaux, schnorr, bampi}@inf.ufrgs.br

Abstract. *This paper presents a detailed energy consumption analysis, considering the energy consumption related to CPU, cache memory and main memory of parallel applications on HPC systems. Furthermore, this paper also presents the correlation between energy consumption, Speedup, and execution time. Experiments are conducted with the NAS parallel benchmarks using three different measurement tools: 1) Intel PCM, 2) Perf Linux, and 3) HP CACTI. The results show a comparison between two approaches to obtain energy consumption results. One using PCM and other using Perf and CACTI. The DRAM results show an average variation between these approaches of 47% for sequential applications, and 19% for parallel applications. The system results show that the lowest energy consumption occurs only when all physical cores are used, showing that the hyper-threading system did not bring benefits in energy consumption to the system. Moreover, the cache memories results show that the cache miss rate (regardless of the level) increases with the number of threads. However, a parallel application has lower cache memory energy consumption when compared to its sequential version.*

1. Introduction

Power and energy are primary concerns in High-Performance Computing (HPC) systems design [1] [2]. Energy efficiency of HPC systems is a critical issue, and to improve that it is necessary to profile power consumption of real systems at a fine granularity [3]. Many factors influence power and energy consumption in high-performance systems, including each component's electrical specification, the system usage characteristics of the applications, and system software [3]. In parallel programming interfaces that use shared memory to communicate during execution (e.g. OpenMP), the communication represents an important part of the total energy consumed by the system [4]. Furthermore, the performance will increase as a result of the system parallelization, promoting energy reductions in the system [4]. However, the increase in performance is not linear and may not scale with the number of threads, due to external bandwidth and the data-synchronization [5].

Power consumption and application performance are coupled and often conflicting and complex [3]. Improving energy efficiency without negatively affecting the performance is challenging [3]. Although there has been several research conducted on

performance and scalability of HPC applications, there is a lack of work that perform a detailed analysis of the energy consumption of the memory hierarchy.

In this paper, we propose a technique that allows us to have a deep understanding of how the energy is consumed by the memory hierarchy. Our technique is based on a mathematical model that is fed by hardware counters from the architecture. The NAS Parallel Benchmarks (NPB) [6] are used for the tests and three profiler tools are used to gather results: Intel PCM [7], Perf Linux [8], and HP CACTI [9]. The Intel PCM provides the energy consumption results for main memory and CPU. The Perf profiler gathers the CPU statistics, monitoring hardware counters. CACTI estimates the memory energy consumption per access.

This paper is organized as follows. Section 2 analyzes the related work. Section 3 describes the analysis methodology. Section 4 presents the results and their evaluation. Finally, Section 5 concludes the paper.

2. Related Work

There has been many research conducted on Speedup and scalability of parallel applications in HPC. The fixed-time speedup model [10] and memory-bounded Speedup model [11] extend the Speedup in one way. However, these metrics focus on performance and ignore both energy consumption and the performance effects on power consumption [12].

Few works investigated the energy consumption impact in shared memory multiprocessing applications. The authors in [13] present the impact of parallel programming models and CPU clock frequency on energy consumption of HPC systems. In this study, the authors evaluated the energy consumption of a parallel system using OpenMP with MPI, using the NAS benchmarks for the experiments. This work showed the influence of different parallel programming models on the energy efficiency and execution performance. The authors in [4] analyzed performance, energy consumption and energy delay product on three embedded processors, using four PPI (Pthreads, OpenMP, MPI-1 and MPI-2). Their experiments showed that OpenMP PPI is more energy efficient than others for the embedded processors and programming environments studied.

The authors in [14] evaluated OpenMP applications on HPC systems and they examined two compilers, along with a variety of algorithm and optimization levels. The experiments in [14] were conducted on a Sandybridge node and they show that the energy consumption varies due to several factors. Jacobson et al. perform in [15] a comparison between a power model widely used in industry with specified types in MCPat tool [16], obtain energy consumed by the computer system.

The aforementioned studies contain relevant and insightful information. However, these solutions lack a detailed memory energy consumption analysis for parallel applications. Besides, they do not provide a relation between energy consumption and Speedup of applications. Thus, our work presents a detailed energy consumption analysis, considering the energy consumption related to CPU, cache memory, and main memory.

3. Methodology

This section presents the employed methodology, which includes some basic concepts about the benchmark and the profiler tools, description of the experimental setup, and the

model used to energy consumption measurement.

3.1. Benchmark Description

The NAS Parallel Benchmarks (NPB) are a set of programs designed to help evaluate the performance of parallel supercomputers [6]. The benchmarks are derived from computational fluid dynamics applications and consist of four kernels and three pseudo-applications specifications. The four kernels are: 1) EP - Embarrassingly Parallel; 2) CG - Conjugate Gradient; 3) MG - Multi-Grid on a sequence of meshes; and 4) FT - discrete 3D fast Fourier Transform. And the three pseudo-applications are: 1) BT - Block Tri-diagonal solver; 2) SP - Scalar Penta-diagonal solver; and 3) LU - Lower-Upper Gauss-Seidel solver.

3.2. Experimental Setup

The workstation used in the experiments is a Sandybridge-based node that has two Xeon E5-2670 processors, each with eight cores. Each processor is clocked at 2.6 GHz, with a peak performance of 166.4 Gflop/s. Each core has 64 KB of L1 cache (32 KB data and 32 KB instruction) and 256 KB of L2 cache. All eight cores share 20 MB of last level cache (LLC), also called L3 cache. The off-chip memory system is composed by two 16GB DDR3 running at 1600 MHz.

In the experiments, 32 threads are used to fill all logical cores of the system (the threads are fixed in each core), with 30 random executions for each configuration of the experiment. These executions were done to get more reliable results since real energy consumption varies with the operating system and other process running concurrently on the machine. Among the benchmark classes, the class B was chosen for the experiments, since it defines applications with larger input size than others. The OpenMP NPB with default configuration for shared memory architectures was used herein.

3.3. Energy Profiler Tools

In this work it is used three tools to obtain the energy measurements: 1) Intel Performance Counter Monitor (PCM) [7], 2) Perf Linux [8], and 3) HP CACTI [9].

The PCM was developed by Intel to study the energy consumption of any application that is executed in recent architectures, such as Intel Xeon, Sandybridge, Ivy Bridge, Haswell, Broadwell, or Skylake processors [7]. If a parallel HPC application is executed in various sockets, PCM will output CPU energy, Dynamic Random Access Memory (DRAM) energy, NUMA details, performance flaws, and so forth in various formats to end users. The tool considers Machine Specific Registers (MSR) using RAPL counters to disclose the energy consumption details of the application [7].

Perf is a profiler tool for Linux-based systems that abstracts away CPU hardware differences in Linux performance measurements [8]. Perf is based on the perf_events interface exported by recent versions of the Linux kernel.

CACTI is an integrated cache and memory access time, cycle time, area, leakage, and dynamic power model [9]. The CACTI cache access model takes in the following major parameters as input: cache capacity, cache block size (also known as cache line size), cache associativity, technology generation, the number of ports, and the number of independent banks (not sharing address and data lines). As output, it produces the cache

Table 1. Memory specification for CACTI simulation

Parameters	Cache L1	Cache L2	Cache L3	DDR3
Capacity	32KB	256KB	20MB	1GB
Block size	64B	64B	64B	64B
Associativity	8	8	16	1
Technology	32nm	32nm	32nm	90nm
Banks	1	1	1	8
Type	cache	cache	cache	main memory
Model	UCA	UCA	NUCA	UCA

configuration that minimizes delay (with a few exceptions), along with its power and area characteristics.

Table 1 presents the main input for CACTI simulation, this specification follows the hardware description presented in the previous subsection. However, it can be observed in Table 1 that the main memory capacity is just 1GB, whereas the Xeon server has 16GB. This happens because CACTI simulator has a memory size limitation of the 1GB. Due to this fact, it is expected that the estimated DRAM results may be inaccurate, but it is expected that they follow the trend.

3.4. Energy Consumption Measurement

The amount of cache loads, stores, load misses, and store misses are generated by Perf tool. This tool provides a list of events to measure micro-architectural events such as the number of cycles, L1 or LLC cache misses and so on. However, this tool does not provide events for intermediate levels of cache, such as L2. To estimate the L2 events it was used the results obtained from L1 and LLC, where L1 misses represents the L2 total accesses and L3 total accesses represents the L2 misses. The energy per access used in the next equations was obtained with CACTI and the specifications presented in Table 1. The energy results per cache access reported by CACTI can be found in Table 2.

Equation 1 presents the formula to estimate the energy consumption for read operations, where the *#hit_rd_cl* are the amount of read hit accesses to cache level (i.e. L1, L2 or L3), the *#miss_rd_cl* are the amount of read miss accesses to cache level, *E_ac_rd_cl* are the energy consumption per read access to certain cache level, and *E_ac_wr_cl* are the energy consumption per write access to certain cache level.

Equation 2 presents the formula to estimate the energy consumption for write operations and follows the same idea aforementioned to calculate the energy for read operations. Both equations 1 and 2 are used to estimate the cache energy consumption for the three cache levels, L1, L2, and L3.

$$EL1_{read} = (Hit_{readL1} \times E\_ac_{readL1}) + (Miss_{readL1} \times (E\_ac_{readL1} + E\_ac_{writeL1})) \quad (1)$$

$$EL1_{write} = (Hit_{writeL1} \times E\_ac_{writeL1}) + (Miss_{writeL1} \times (E\_ac_{readL1} + E\_ac_{writeL1})) \quad (2)$$

The total cache energy consumption, representing the sum of the energy consumed in the three levels of cache memory, is summarized in Equation 3. The CPU and DRAM

Table 2. CACTI estimated energy consumption per access

	Cache			
	L1	L2	L3	DRAM
Energy Access	$0.0164nJ$	$0.0731nJ$	$0.567nJ$	$5.603nJ$
Static Power	$0.011W$	$0.085W$	$0.86W$	$1.45W$

Table 3. Execution time and energy consumption results for NPB

NPB	**Time** (s)				**CPU Energy** (J)				**DIMM Energy** (J)			
	1	16	24	32	1	16	24	32	1	16	24	32
BT.A	61.4	5.4	6.9	5.5	3184.6	756.0	926.3	838.8	322.6	40.6	49.5	45.2
CG.A	1.7	0.4	0.5	0.5	126.9	69.7	74.9	71.4	14.8	7.3	7.6	7.5
FT.A	6.2	0.7	0.8	0.8	356.8	131.9	139.7	139.0	38.1	12.3	13.0	12.9
LU.A	49.1	5.0	5.8	4.8	2535.4	673.4	789.2	691.4	261.1	44.8	50.5	47.9
MG.A	3.6	1.1	1.2	1.3	225.2	154.7	167.1	179.6	27.0	15.9	17.7	18.9
SP.A	43.9	4.9	5.9	5.2	2273.8	652.2	785.7	749.0	253.1	46.1	53.2	54.2
BT.B	264.3	32.7	37.5	34.1	13530.7	4029.4	4467.6	4320.5	1380.7	287.0	314.9	305.1
CG.B	117.0	28.8	28.9	29.4	5954.2	2974.3	2946.0	3077.7	673.6	254.8	266.1	276.5
FT.B	74.0	8.3	9.3	8.4	3835.2	1101.2	1153.9	1152.8	400.0	84.5	91.6	90.4
LU.B	211.6	27.3	30.3	28.2	10764.5	3411.1	3876.2	3716.5	1132.7	257.5	280.7	274.2
MG.B	12.3	3.6	3.9	4.1	675.6	432.7	474.5	514.6	79.7	41.1	46.6	51.0
SP.B	210.0	50.5	53.3	51.5	10704.2	5751.8	6115.9	6152.4	1251.7	524.3	544.3	559.7

energy results were obtained by Intel PCM tool.

$$Cache_{TotalEnergy} = EL1_{read} + EL1_{write} + EL2_{read} + EL2_{write} + EL3_{read} + EL3_{write} + Power_{Static} \times Time \quad (3)$$

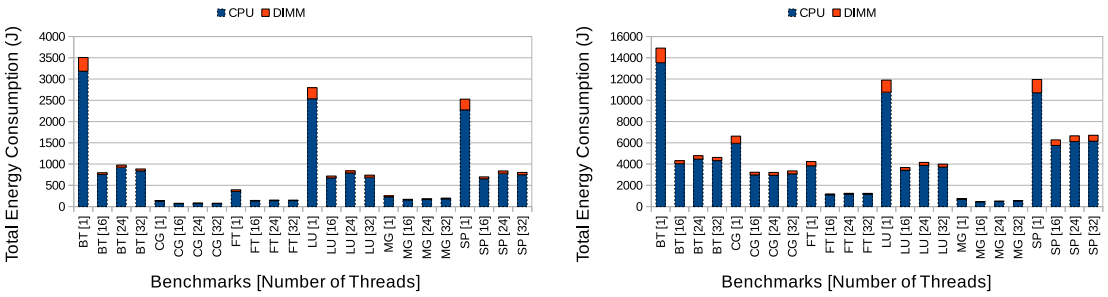
4. Results and Discussions

In this section, the results of the experiments are discussed. Subsection 4.1 presents and discuss the DRAM and CPU energy consumption for the benchmarks. Subsection 4.2 shows the Speedup and energy-efficiency results. Subsection 4.3 presents a comparison of the DRAM energy consumption results obtained by two energy estimation tools. Subsection 4.4 presents the detailed results of the cache memory energy consumption.

4.1. System Energy Consumption

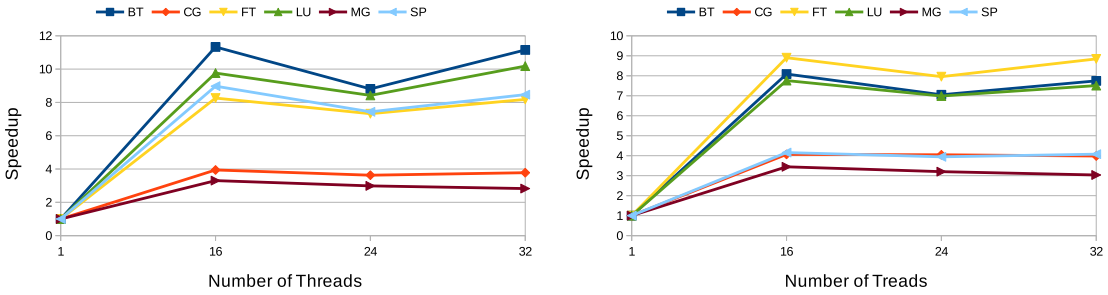
Table 3 presents the energy consumption results obtained with the PCM tool for all applications. These results were obtained from the sequential version and three parallel versions with 16, 24, and 32 threads. The DIMM Energy is the main memory's energy, the CPU Energy is the total energy consumption of the processors (32 cores) with different cache levels: L1 (inside the core), L2 (outside the core and inside the chip) and L3 (socket level).

It can be seen in Table 3 that the execution time is directly related with the energy consumption. For the same applications such as "CG.A" and "CG.B" (different input size) there is a considerable increase in the execution time (67 times more), as well as in the energy consumption (46 times more). It is possible to see this characteristic in all applications presented in Table 3. Furthermore, it can be seen in this table that



(a) Class A (b) Class B

Figure 1. CPU and DRAM energy consumption results of the benchmarks



(a) Class A (b) Class B

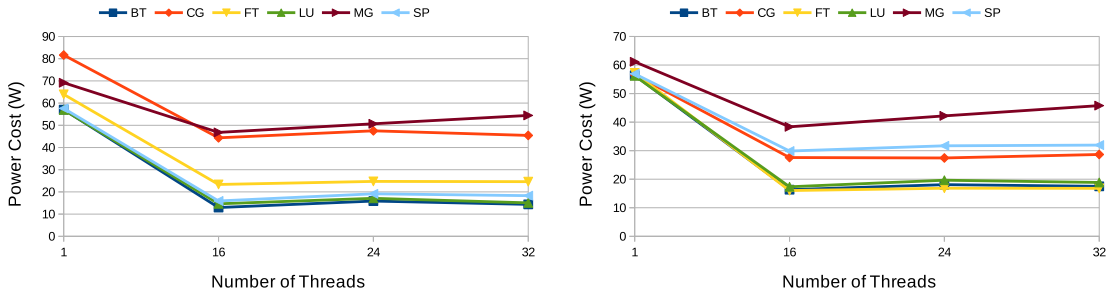
Figure 2. Speedup results for all benchmarks

approximately 90%, on average, of the total energy consumption corresponds to the CPU, including caches memories, and the remaining to DRAM. This occurs for all applications, showing that the energy spent in processing is a major bottleneck for these benchmarks.

Figure 1 presents the energy consumption results for CPU and DRAM obtained with the PCM tool for all applications. It is possible to observe that the applications BT, LU and SP present higher energy consumption than others applications. This happens because these applications present the higher problem sizes, i.e. number of iterations, grid size, and time step.

The results presented in Figure 1 and Table 3 show that when the parallel versions of the applications are used the energy consumption is reduced significantly, from 30% to 70% of reduction. This happens because the parallel version, even using more computational resources, has a much smaller execution time that sequential version. This only happens because these benchmarks present a high level of parallelism.

It can also be seen in Figure 1 that the lowest energy consumption occurs when the 16 physical cores are used, showing that the hyper-threading system did not bring any benefits in energy consumption. This becomes more evident when Figure 2 is observed; this figure presents the Speedup results for all benchmarks and it is explained in details in the Subsection 4.2.



(a) Class A (b) Class B

Figure 3. Power-Cost results for all benchmarks

4.2. Speedup and Performance

Figure 2 presents the Speedup results for all benchmarks. In this figure, it can be seen that the best Speedup for all benchmarks occurs when the 16 physical cores are used. The Class A best Speedup occurred in the BT (Block-Tridiagonal) application, this application has a maximum Speedup $11.3\times$ faster than sequential version, using 16 threads. However, the BT application did not have a good scalability, since it reached a Speedup of $8\times$ in Class B.

Meanwhile, the FT (Fourier Transform) application presented a Speedup of the $8.3\times$ in Class A and $8.9\times$ in Class B, reaching a good scalability. Furthermore, this application presented the highest Speedup among all benchmarks in Class B. This occurs because this application contains the computational kernel of a 3-D fast Fourier Transform, performing three one-dimensional (1-D) FFT's, which presents high degree of parallelism.

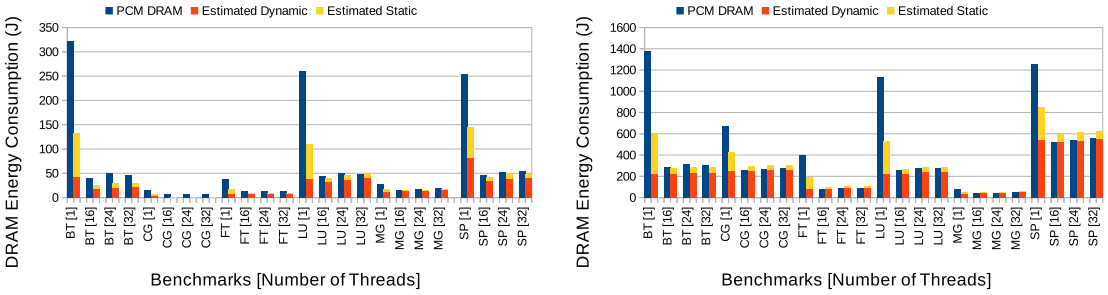
Nonetheless, the MG (Multi-Grid) application did not scale as well as the others, since it applies a multi-grid on a sequence of meshes, performing memory intensive accesses. Besides, this application represents a small problem and presented the lowest energy consumption among applications, as show Figure 1.

A correlation between energy consumption, Speedup, and execution time is presented in Figure 3. In this figure the power-cost results are presented, which are obtained by Equation 4. With these results it is possible to check if an application is energy efficient regardless of the input size.

$$PowerCost = \frac{Power}{Speedup} \tag{4}$$

As can be seen in Figure 3, applications BT, SP, and LU presented the best power-cost results for benchmarks Class A, while MG had the worst result. In the Class B benchmarks, the applications FT, BT, and LU presented the best power-cost results and MG presented the worst result again, even being the application that had the lowest energy consumption, showing that this application does not present a good level of parallelism.

Moreover, it is possible to see in Figure 3 that BT and LU applications presented low variation in power cost for both classes. BT varies from $13W$ to $16W$ and LU varies



(a) Class A

(b) Class B

Figure 4. Results of the DRAM energy consumption estimation for the benchmarks

from $14.6W$ to $17.3W$. However, the FT and SP applications showed a high variance while the FT reduced the power cost from $23.3W$ to $16W$ the application SP increased the power cost from $16W$ to $30W$. These facts are strongly related to the scalability presented by these applications and previously discussed.

4.3. Comparison of the DRAM Energy Consumption Results

Figure 4 presents the DRAM energy consumption results obtained by CACTI tool (estimated) and the real energy consumption obtained by PCM tool for different applications, using 1, 16, 24 and 32 threads.

The CACTI tool provides the average energy consumption per access for both caches and DRAM memories, in accordance to the specified platform. The calculation of the estimated energy consumption was performed using the Equation 5.

$$DRAM_{TotalEnergy} = Total_{ReadWriteAccesses} \times Energy_{Access} + Power_{Static} \times Time \quad (5)$$

The applications that consumed more energy from DRAM memory are those that have a high rate of cache misses when the miss occurs on the last level of the cache, which implies in accesses to the DRAM memory. The applications that presented high energy consumption of the DRAM memory, also presented high energy consumption in the last level cache. Another interesting point is that applications that had a high energy consumption in the last level cache exhibit the same behavior in the energy consumption of DRAM, as an example of this behavior we have BT applications and CG.

The applications, when executed in parallel, presented lower DRAM energy consumption when compared to the sequential version. For example, the estimation of the theoretical energy consumption for applications sequential benchmark is on average 4-fold lower than the actually estimated. As for the parallel application, the approximation of the theoretical estimate to the actual increases, being about 2 times lower than the actually estimated on average. This behavior can be explained by considering only the dynamic energy to the calculation, which could approximate the theoretical and the actual estimate for sequential applications when evaluating this behavior for DRAM.

Considering the dynamic and static energy for DRAM, both estimated by CACTI, we can see that for sequential applications the theoretical estimate is closer to the real,

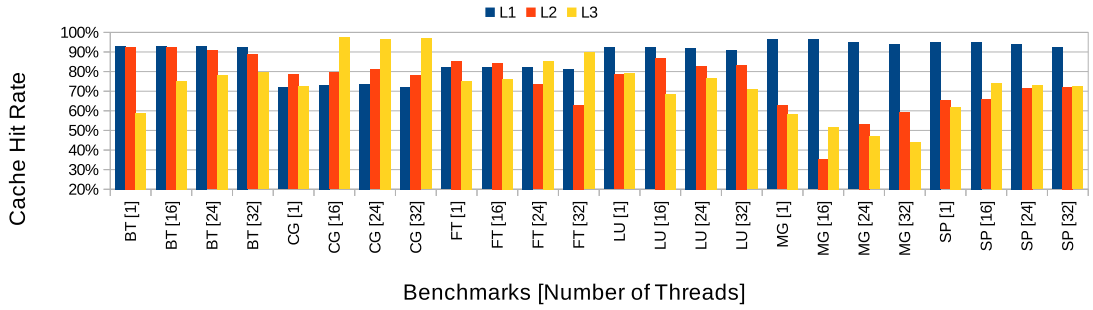


Figure 5. L1, L2 and L3 cache hit results for benchmarks class A

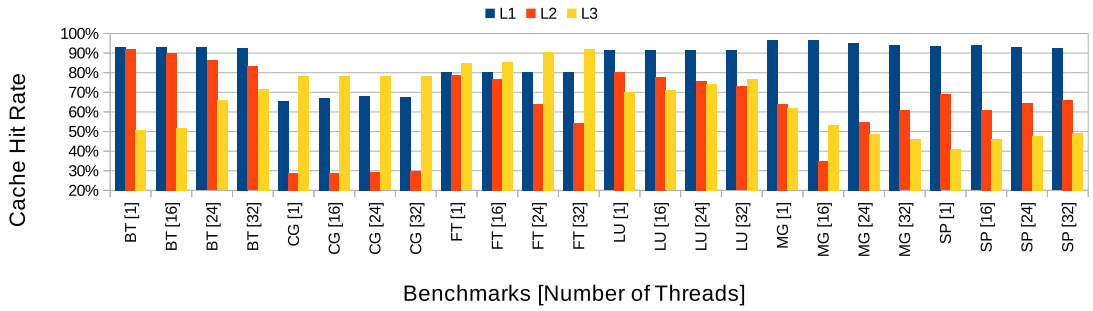


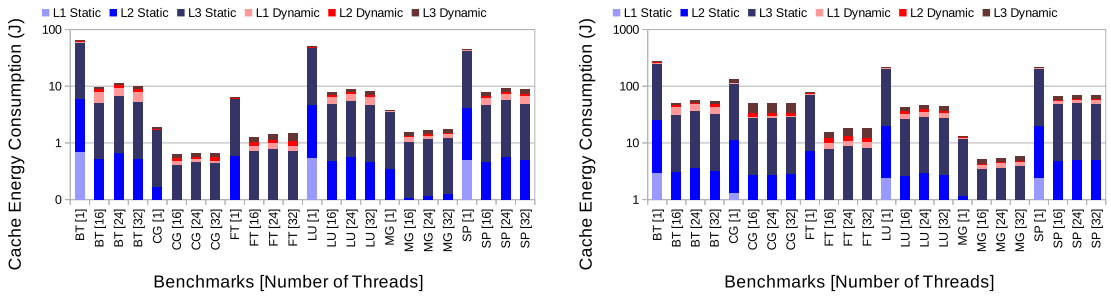
Figure 6. L1, L2 and L3 cache hit results for benchmarks class B

averaging $2\times$ lower, increasing accuracy (4 times less considering only dynamic energy). As for the parallel application also occurs an improvement in theoretical estimation, being about $1.6\times$ lower than the actually estimated (2 times less considering only the dynamic energy). From these results, we can see the efficiency of the CACTI tool for DRAM estimation of power consumption.

4.4. Cache Memory Energy Consumption

Profiler tools could not report the energy consumption results for cache memory individually, since the CPU is a closed architecture and the data tend to be confidential. Thus, the only information that can be extracted with these tools is the number of accesses to the hardware counters and the total energy consumption of the CPU. However, in this paper we present an estimated energy consumption for the cache memories, and these results are presented and discussed in this subsection.

Figures 5 and 6 present the hit rate results for the three levels of the cache memory. These results were obtained from the hardware counters using the Perf tool. It can be seen in these figures that the hit rate decreases when class B is used, thus the miss rate increases, increasing the amount of access to the DRAM. The CG and MG applications presented very different results from the others applications. The CG application presented L3 hit rate results higher than other levels in both classes. This is due to irregular accesses performed to the memory, despite the irregular accesses, the accesses are in nearby regions. This behavior did not occurs on MG application, which also performs irregular accesses to the memory, but the accesses are not performed on neighboring regions, since this application has the lowest L3 hit rate, which is



(a) Class A (b) Class B

Figure 7. Energy consumption results of the cache memories for all benchmarks

approximately 45%.

Figure 7 presents the estimated energy consumption of the cache memories. These results were obtained following the Equations 1, 2 and 3 presented in Section 3.4 with Perf Linux and HP CACTI tools. The Perf tool uses the hardware counters to generate the amount of accesses and the CACTI generates the energy consumption per access.

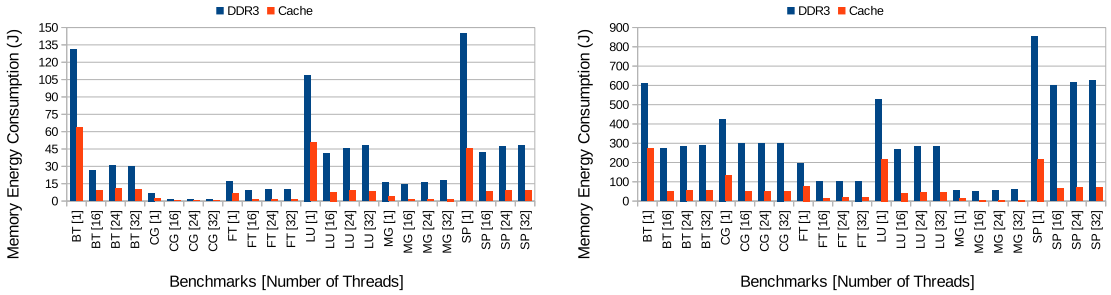
Similar to what happens with the DRAM and CPU energy consumption results, the cache energy consumption reduced as more the number of threads increase. This fact is mainly due to reducing in static energy, for example in class A the BT sequential application consumed $4.3J$ of dynamic energy and $59J$ of the static energy, i.e. the static energy is $13.8\times$ higher than the dynamic energy in this case. This same application running with 16 threads has a dynamic energy consumption of $4.2J$ and $5.2J$ of the static energy, reducing $11\times$ the static energy consumption while the dynamic consumption is almost the same. This behavior is maintained for the Class B benchmarks.

It can be also observed in Figure 7 that the L3 total energy consumption is greater than the L1 and L2 energy consumption. This is due to the higher cost per access in level L3, the highest miss rate and the high static energy consumption presented in this level. Considering only the dynamic energy consumption, the applications BT, LU and MG presented higher L1 energy consumption results, showing that these applications have better reuse of data and have a good locality to get the best performance.

Figure 8 shows the estimated energy consumption results for the three levels of cache memory and DRAM memory. It can be seen in Figure 8 that the benchmarks from class B have higher energy consumption, this is due to the fact that class B has larger inputs which generates high computation time, leading to greater energy consumption.

Figure 8 also shows that in all applications the energy consumed by DRAM is greater than the energy consumed by the cache memories. In Class A applications, the biggest difference is found in sequential MG application, which has an energy consumption of DRAM $9.4\times$ greater than the energy consumption of the cache; DRAM consumed $16.1J$ and the caches consumed $1.7J$. The smallest difference in Class A is found in CG application with 16 threads. In this application the difference is only $1.7\times$, where the DRAM consumed $1.06J$ and the caches consumed $0.63J$.

In class B, the biggest difference is found in the MG application with 32 threads,



(a) Class A

(b) Class B

Figure 8. Comparison between DRAM and cache energy consumption results to the benchmarks

this application presented a DRAM energy consumption $10\times$ greater than the cache energy consumption; in this application the DRAM consumed $58J$ and the cache consumed $5.8J$. The smallest difference, $2.2\times$, is found in the BT application. In this application, the DRAM consumed $609J$ and the cache $272J$. It is observed that with the increase in the applications input; the difference between DRAM and cache energy consumption have a significant increase, since there is an increase in the miss rate in the last level cache, generating more accesses to the DRAM.

5. Conclusion

This work presented and analyzed the energy consumption results of the CPU and memory system for shared memory parallel applications on High-Performance computers. The NAS Parallel Benchmarks was used for the tests and three tools were used to generate the results: PCM, Perf, and CACTI. The results showed that the benchmarks achieved good results for parallelization on a shared memory system. Furthermore, the energy consumption estimation for the cache memories and the power-cost metric contributes to a detailed analysis of the application. The results also showed that the lowest energy consumption occurs only when all physical cores were used, showing that the hyper-threading system did not bring benefits in energy consumption to the system. Moreover, the cache memories results showed that the parallel applications presented lower energy consumption in the cache memory when compared to its sequential version, despite the increase in cache miss rate generated by the threads.

As future work we will investigate the memory energy consumption for others applications, such as video encoders. Such work will consider multithreaded embedded architectures, as battery-powered devices have stringent energy restrictions.

Acknowledgment

This work was partially financed by the National Council for Scientific and Technological Development (CNPq), Coordination of Improvement of Superior Education Staff (CAPES), Research Support Foundation of Rio Grande do Sul (FAPERGS), and MCTI/RNPBrazil under the HPC4E Project, grant agreement n° 689772.

References

[1] J. Mair, Z. Huang, D. Eyers and Y. Chen, "Quantifying the Energy Efficiency Challenges of Achieving Exascale Computing", IEEE International Symposium on Cluster, Cloud and Grid Computing, pp. 943-950, 2015.

[2] R. Gioiosa, D. Kerbyson, and A. Hoisie, "Evaluating performance and power efficiency of scientific applications on multi-threaded systems", International Workshop on Energy Efficient Supercomputing, pp. 11-20, 2014.

[3] R. Ge, X. Feng, S. Song, H. C. Chang, D. Li and K. W. Cameron, "PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications", IEEE Transactions on Parallel and Distributed Systems, vol. 21, no. 5, pp. 658-671, 2010.

[4] A. F. Lorenzon, A. L. Sartor, M. C. Cera and A. C. S. Beck, "Optimized Use of Parallel Programming Interfaces in Multithreaded Embedded Architectures", IEEE Computer Society Annual Symposium on VLSI, pp. 410-415, 2015.

[5] M. A. Suleman, M. K. Qureshi and Y. N. Patt, "Feedback-driven threading: power-efficient and high-performance execution of multi-threaded workloads on CMPs", International Conference on Architectural Support for Programming Lang. and Oper., pp. 277-286, 2008.

[6] "NAS Parallel Benchmarks", http://www.nas.nasa.gov/publications/npb. html, June 2016.

[7] "Intel Performance Counter Monitor - A better way to measure CPU utilization", http://www.intel.com/software/pcm, June 2016.

[8] "Linux Perf tool", https://perf.wiki.kernel.org/, June 2016.

[9] "CACTI 6.5", http://www.hpl.hp.com/research/cacti, June 2016.

[10] J. L. Gustafson, "Fixed Time, Tiered Memory, and Superlinear Speedup", Distributed Memory Computing Conference, pp. 1255-1260, 1990.

[11] X.-H. Sun and L. M. Ni, "Scalable problems and memory-bounded speedup", Journal of Parallel and Distributed Computing, vol. 19, pp. 27-37, 1993.

[12] S. Song, C. Y. Su, R. Ge, A. Vishnu and K. W. Cameron, "Iso-Energy-Efficiency: An Approach to Power-Constrained Parallel Computation", IEEE International Parallel & Distributed Processing Symposium, pp. 128-139, 2011.

[13] J. Balladini, R. Suppi, D. Rexachs and E. Luque, "Impact of parallel programming models and CPUs clock frequency on energy consumption of HPC systems", International Conference on Computer Systems and Applications, pp. 16-21, 2011.

[14] A. K. Porterfield, S. L. Olivier, S. Bhalachandra and J. F. Prins, "Power Measurement and Concurrency Throttling for Energy Reduction in OpenMP Programs", International Parallel and Distributed Processing Symposium Workshops, pp. 884-891, 2013.

[15] H. Jacobson, P. Bose, G. Wei, and D. Brooks, "Quantifying Sources of Error in McPAT and Potential Impacts on Architectural Studies", in 21st International Symposium on High Performance Computer Architecture (HPCA), 21st. IEEE, 2015.

[16] "McPAT", http://www.hpl.hp.com/research/mcpat, June 2016.