26th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing

Performance and Energy Consumption Analysis of Coprocessors using Different Programming Models

Robson Gonçalves, Alessandro Girardi, Claudio Schepke

Federal University of Pampa (UNIPAMPA), Alegrete Technology Campus – Alegrete – RS – Brazil
robsongoncalves,alessandrogirardi,claudioschepke@unipampa.edu.br

***Abstract*—The optimization of the relation between performance and energy consumption is a strong requirement mainly in high performance environments. The top 10 Green500 supercomputers use accelerators/coprocessors as primary approach to increase the performance while reducing energy consumption. This paper presents a study on the main factors that impact this relationship, evaluating and comparing Intel programming models on an Intel Xeon Phi coprocessor architecture. The methodology applied in this work consists of evaluating performance and energy consumption on execution scenarios using Linpack and HPL 2.1 benchmarks. These scenarios consider various environment parameters and execution on the Intel *host*, *offload* and *native* programming models. Experimental results indicate that the *host* and *offload* models are more efficient in the performance per energy consumption relationship with shared memory and distributed memory, whereas the native model demonstrated better efficiency in energy consumption.**

I. Introduction

The increasing in the performance and computational capacity is associated, in general, to the increasing in energy consumption, mainly in data centers or large environments. In this scenario, the processor industry has presented models based on heterogeneous architectures composed of *hosts* with multicore CPUs connected via PCIe with manycore graphic accelerator units, enabling low-cost processing in the order of teraflops [20]. To support this coprocessor-based architecture, Intel also provides the Many Integrated Core (MIC) architecture and programming models such as *host*, *offload* and *native*, to provide flexibility and benefits for running applications in environments which require high performance.

There are a number of papers that address the performance and energy relationship of Xeon Phi architecture [6], [8]. The work in [21] compares Xeon Phi, GPU and FPGA on multicore systems. The analysis and comparison of results on the impact of the number of threads in Xeon Phi coprocessors is presented in [18]. The work in [8] describes the monitoring of energy consumption and performance on Nvidia Tesla GPUs and Intel Xeon Phis. However, there is a still need for further studies in optimization and better use of energy, especially in the programming models.

The goal of this paper is to evaluate the main configurations and environment parameters that are directly associated to performance and energy consumption of Intel Xeon Phi coprocessors for the available programming models. To support this analysis, several behaviors related to performance and energy consumption are tested during the execution of the Linpack benchmark for shared memory machines and the HPL 2.1 for distributed memory environments. As contribution, this work identifies which models are more efficient in terms of energy consumption without compromising performance in high performance environments. The analysis on the impact of energy consumption in shared and distributed memory environments of MIC architectures allows to identify which parallelism factors - like number of nodes, cores, processes, threads, workload size, among others - have relation in energy and performance impact.

II. Related Work

The work of [1] demonstrates a detailed model of instruction-level energy consumption on Intel Xeon Phi processors. This model provides software developers the opportunity to improve energy efficiency in their codes with energy savings of up to 10% due to redundant custom *prefetch* operations in Linpack's custom implementation. Heterogeneous architectures have grown greatly with the appearance of *manycore*. Another technique used for performance optimization is the use of Thread Affinities. The work [6] use this approach to investigates the effects of varying options in thread affinities on Intel Xeon Phi coprocessors, considering the use of cores to evaluate energy consumption and run time.

The work performed by [2] presents a study to understand the best balance between performance and energy through the customization of use of processing in hybrid computing applications composed by CPU+GPU and CPU+Xeon Phi. This work also used RAPL interface for monitoring CPU power and *micsmc* for power in Xeon Phi. Another work related to hybrid and manycore architectures is [3], where it presents MT-MPI proposal, an implementation for coordination of multithreads internally in MPI together with OpenMP.

The DVFS (Dynamic Voltage and Frequency Scaling) is a widely used technique to save energy consumption, in which the CPU frequency and supply voltage is manual or automatically adjusted [8]. The work of [4] presents a compiler to optimize programs in order to identify regions where the code can be reduced without compromising performance. The evaluation of different clock frequencies of CPUs for efficiency in energy consumption over programming paradigms, using OpenMP on shared memory and MPI for distributed memory architectures is presented in [5].

There are also several academic studies on NVIDIA GPUs for evaluating performance and CUDA-based programming models. A comparative was done by [8], which describes the proposal of a scheme called EEA (Efficiently Energetic

0-7695-6380-5/18/$31.00 ©2018 IEEE
DOI 10.1109/PDP2018.2018.00086

Acceleration) that captures and controls data in real time by means of two monitors called enerGyPU (for GPU) and enerGyPhi (for Xeon Phi).

The E-Team software proposed by [16] is a mechanism based on RAPL to monitor and account the energy consumption of individual applications and multi-core environments and can be used as a service with the possibility of starting and stopping when necessary. To evaluate the impact of the number of threads in Intel Xeon Phis, the work in [18] proposes a design tool to monitor and record the energy usage in real time during execution. The results can be compared for different executions and different numbers of threads.

There are some works on external power meter comparisons. In [17] is proposed GreenHPC, an external framework of components for use in HPC environments. The work in [21] shows a study that involves GPU, Xeon Phi and FPGA co-processors to embrace the vast majority of hardware accelerators applied in modern HPC systems.

Recent works related to energy efficiency in Intel Xeon Phi Knight Landing processors are presented, such as [7]. It aims to set the power limits according to the workload characteristics and application performance. This work shows how the variation on the Xeon Phi operating frequency influences the energy consumption and performance.

III. Performance and Energy Evaluation

The Intel Parallel Programming Models are divided into two groups. The first is the *offload* mode, in which parallelism is explicit using shared virtual memory and basically for calls using *OpenMP* through the use of *runtime offload libraries* in *MPSS*. The second is the *native* that can be executed in standalone applications (compiled for *mic* architecture) or applications distributed through MPI processes that use ssh connections and TCP/IP protocols between the host and Intel Xeon Phi coprocessors. The *host* model is used for serial processing or centralized parallelism in *host* whose behavior demands higher memory latency. In the *symmetric* model, both hosts and coprocessors execute the same HPC application consisting of several processes using standard communication through MPI processes [13], [11]. The *native* model is a variant of the *symmetric* model, allowing the application to execute one or more processes on the coprocessor, not depending on the *host*, but only on the SCIF and VEth (Virtual Ethernet) drivers [13]. The experiments were done using a server with two CPUs and four Xeon Phi, as shown in Table I.

A number of benchmarks such as NAS (EP, FT, IS, MG), Sparse, Matrix Multiplication, among others have been used for performance evaluation of processors and coprocessors. In this work the most relevant criteria of choosing the benchmarks are the high degree of relevance to the Top500 ranking, the compatibility and support for shared memory in heterogeneous environments and the use of distributed memory through MPI and OpenMP. In this context, the executions were performed using the benchmarks *Linpack* and *HPL 2.1*. Linpack uses BLAS (Basic Linear Algebra Subprograms) package that provides a set of routines to work with linear algebra operations such as vector addition, scalar multiplication, linear combinations, and matrix multiplication. The Highly-Parallel Linpack (HPL) benchmark solves dense systems of linear equations by partial Gaussian elimination with partial pivoting [8], [19].

Intel Open Source RAPL and Power Cap Framework were used for the monitoring and collection of the average power consumption for both CPU and RAM. The available power counters can be accessed at */sys/class/powercap/intel-rapl/*. The monitoring of temperature and memory and core usage of the Intel Xeon Phi coprocessors, is made by the software tool *micsmc*. The energy consumption of the CPU and RAM are calculated by averaging the values recorded in Joule/s in the powercap counters. The energy consumption of the Xeon Phi coprocessors is calculated by averaging the values recorded in the logs generated by the *micsmc*.

For each processing, the configuration parameters are read from the scenario files and shell scripts configure the environment and execute the benchmark. During this execution, a batch process monitors the energy consumption of the host CPUs and DRAMs and the Intel Xeon Phi coprocessors. At the end of this process the files containing benchmarks results and logs of the parameters such as temperature and energy consumption of CPUs, memories and coprocessors are available. The programming models supported by the implemented environment are presented in Table II.

The executions are performed using the parameters described in the Table III. The wide range of values defined in each parameter is necessary because it allows the complete analysis of the behavior of each model in the most varied situations. In this paper only the most relevant parameters and results related to the performance and energy consumption relation are presented.

The metric adopted to measure and evaluate performance is *GFlops/s*, which is the standard used by the Top500. For energy efficiency, there are several metrics proposed in academia

TABLE I. Specification of the experimental platform

	CPU	**Coprocessor**
Host/Device	2 X E5-2699v3	4 x Xeon Phi 7120P
Number of physical cores	2 x 18	4 x 61
Nominal Frequency	2.3 GHz	1.2 GHz
Memory Size	128 GB	16 GB
Threading API	OpenMP	OpenMP
Operation System	Linux kernel 3.10.0-514.26.2.el7.x86_64	

TABLE II. Programming Models supported by the environment.

Programming Model	**Resources**
hosted (CPU)	*CPUs*
offload + 1 Xeon Phi (OF1D)	*CPUs + mic0*
offload + 2 Xeon Phi (OF2D)	*CPUs + mic0 + mic1*
offload + 3 Xeon Phi (OF3D)	*CPUs + mic0 + mic1 + mic2*
offload + 4 Xeon Phi (OF4D)	*CPUs + mic0 + mic1 + mic2 + mic3*
native 1 Xeon Phi (N1D)	*mic0*

TABLE III. Range of scenario parameters.

Parameter	**Range**
align	{4, 8}
cores	{1, 2, 4, 8, 12, 16, 24, 32, 64, 72, 244}
thread affinity	{compact, scatter, balanced}
processes	{2, 4, 8, 16}
number of threads	{8, 12, 16, 24, 32, 64, 72}
workload size	[1000, .., 40000]

and industry. Sun Microsystems introduced the SWaP (Space, Wattage and Performance) metric that assesses the efficiency and effectiveness of rack-optimized server deployments in a datacenter [5]. Another metric is the Energy-Delay Product (EDP) that calculates the amount of energy consumed over a given time frame [14]. The energy metric used in this work is the same as that used by Green500, the *Flops/W* [15]. The parts directly to be evaluated in the energy consumption should be CPU, Memory and Xeon Phi Coprocessors. The total energy consumption is obtained by the partial sum of energy consumption of these components. So, the following equation can be applied to evaluate energy consumption per node of the components [8]:

$$Energy_{Node} = \sum_{j=1}^{nc} P^{j}_{CPU} + \sum_{k=1}^{ng} P^{k}_{XeonPhi} + \sum_{m=1}^{nm} P^{m}_{RAM} \quad (1)$$

Where, P^{j}_{CPU}, $P^{k}_{XeonPhi}$ and P^{m}_{RAM} are the Energy consumption for the CPU, Xeon Phi and memory element, respectively. The variable *nc* is the number of CPUs, *ng* is the number of Xeon Phi coprocessors and *nm* is the number of memory elements. For each scenario execution, the effective energy consumption of these components is the registered average of the consumption for CPU, RAM and Xeon Phi coprocessors, in Watts, during the execution of the benchmark.

IV. Results

This topic presents the preliminary results obtained with the execution of several scenarios created by the combination of the parameters defined in Table III and the models supported by the environment (Table II). With these results it is possible to identify the impact of *host* (CPU), *offload* (OF1D, OF2D, OF3D and OF4D) and *native* models of the behavior obtained considering the different ranges of values presented in the Table III. The results of the Linpack and HPL benchmarks were generated considering number of threads equal to 64, *thread affinity* with *compact* value and *align* equal to 4. The graphs on the HPL benchmark were also generated using the P parameter equal to 1, Q equal to 4 and block size as follows: OF1D = 960 OF2D = 1024 OF3D = 1024.

The performance obtained during the execution of Linpack in shared memory is presented in the Figure 1. It is possible to notice that the models based on *CPU* and *offload* (OF1D, OF2D, OF3D and OF4D) presents a linear increasing in performance for workload with sizes smaller than 10,000. From this point, the performance tends to stabilize. There is no performance gain for different offload configurations. For the *native* model (N1D), the performance behavior follows the same pattern, but with a lower value than the others. The *hosted* and *offload* models demonstrate great similarity in the results. This is because the host memory capacity is shared with the coprocessors in use. In the *native* model, it is limited to only 16GB for each coprocessor used in processing. The use of processing in the *native* mode is suitable for highly parallelized or vectored executions but does not require great memory usage.

The results obtained for the execution of the HPL benchmark are presented in Figure 2. It is possible to show that the *host* (CPU) and *offload* based models performed similarly, with linear growth according to the workload. However, the absolute performance are below the Linpack. For a workload size 10,000 the execution of Linpack obtained more than 750 GFlops whereas in HPL the result was less than 150 GFlops. Similar to the work of [8], the best performance was obtained using thread affinity parameter configured as *compact*. The results presented in the work of [2], which uses the *offload* model and matrix multiplication and fractal benchmarks, demonstrate that the greater the CPU utilization the greater the energy consumption. By the other side, when using the Xeon Phi it occurs the opposite - the longer the CPU processing the smaller the energy consumption.

The results for energy consumption, considering the Linpack shared memory, are presented in Figure 3. The offload model with 4 Xeon Phis (OFD4) demonstrates higher energy consumption in the *CPUs* and very little in the coprocessors. The total energy consumption grows quadratic with the workload size. The *offload* model is not efficient in the performance and energy relationship in shared memory environments because it requires strong CPU dependency on the host for managing the main processing. In the *native* model (N1D), as shown in Figure 4, the energy consumption is significantly lower compared to the other models and it is also proportional to the processing load (workload size).

Figure 5 considers the number of threads and analyze the results obtained with Linpack in shared memory. It is possible to verify that in the models *host* (CPU) and *offload* (OF1D, OF2D, OF3D and OF4D) the energy consumption is similar regardless of the number of threads. The native model (N1D), with the increase in the number of threads, presents decreasing in energy consumption and with 64 threads presents

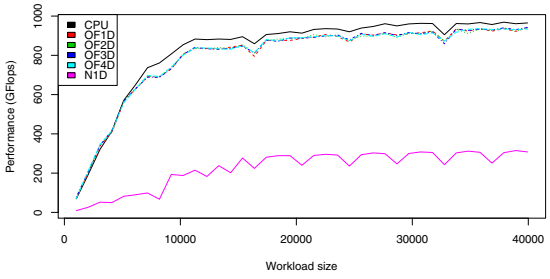

Fig. 1. Performance in shared memory (Linpack).

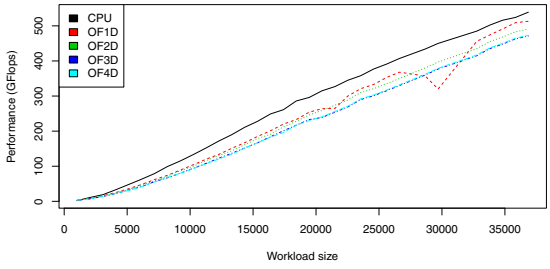

Fig. 2. Performance in shared memory (HPL 2.1)

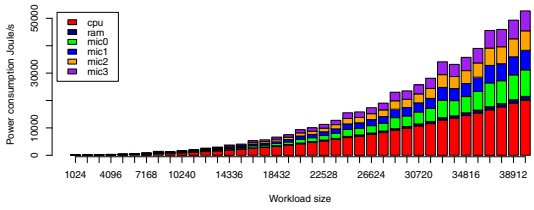

Fig. 3. Energy in *offload* mode with 4 Xeon Phis (Linpack)

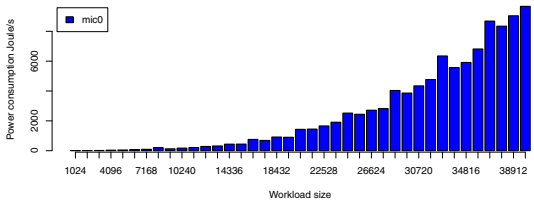

Fig. 4. Energy in *native* mode with 1 Xeon Phi (Linpack).

a significant lower consumption compared to the other models. It demonstrates that the applicability of Xeon Phi coprocessors is recommended for applications with high level of parallelism and low memory processing usage.

The results obtained with the HPL benchmark on distributed memory, presented in Figure 6, show that the energy consumption of *offload* models is much higher than others. The offload model OF3D (three Xeon Phis) presented higher consumption and even higher than OF4D (four Xeon Phis). This demonstrates that the addition of coprocessors will not necessarily raise the performance and energy consumption. Another relevant feature is that in the *offload* model, the scenarios with 3 Xeon Phis (OF3D), with 16 or 32 threads obtained a standard in the energy consumption. But with 64 threads the consumption was larger compared to the scenarios of 1, 2 and 4 Xeon Phis. The *hosted* model was the most energy-efficient during HPL benchmark runs in all scenarios even considering different number of threads.

As a general comparison of energy consumption between Linpack and HPL, it is noticed that due to the great difference of execution time between both benchmarks, the HPL generated the higher energy consumption. In the comparison between the Figure 5 and Figure 6 it is clear that there is an increase in energy consumption when increasing the number of coprocessors. Comparing the energy consumption between the results of the benchmarks, it can be seen from Figure 6 that the energy consumption for 64 threads in the HPL started from values above 10^5 Joules/s, well above the maximum consumption obtained in Linpack (Figure 3), that was close to $5 \cdot 10^4$ Joules/s. It happens mainly due to the high execution time of the HPL benchmark.

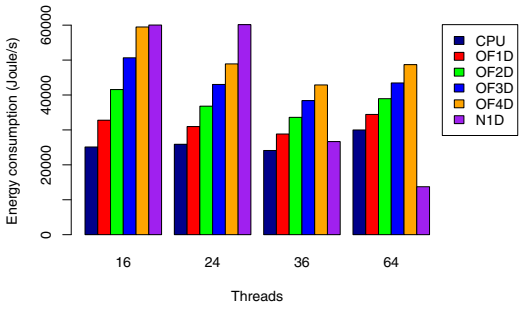

Fig. 5. Energy consumption for different number of threads (Linpack).

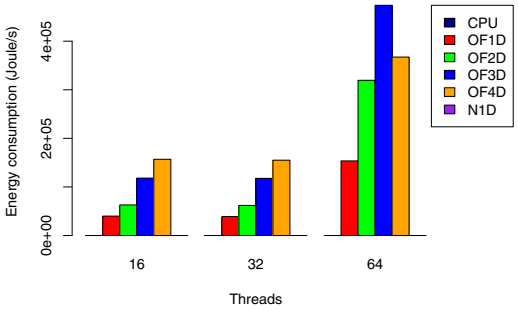

Fig. 6. Energy consumption for different number of threads (HPL).

The execution time is one of the factors that impact on the performance and energy consumption. In the Linpack results presented in Figure 7, it is possible to verify that, following the same behavior presented in the Figure 1, the *host* and *offload* models have execution times very similar, while in the native model (N1D) the execution time is much longer and increases according to the size of the workload. This behavior of longer runtime in the *native* model was also obtained in the results obtained by the HPL benchmark.

Applying the *performance per watt* (PPW) metric (Figure 8), it is possible to identify that the *native* model (N1D) demonstrates the smaller value compared to other models over Linpack benchmark. Among the scenarios of the *offload* model (OF1D, OF2D, OF3D and OF4D), the range was between 1.5

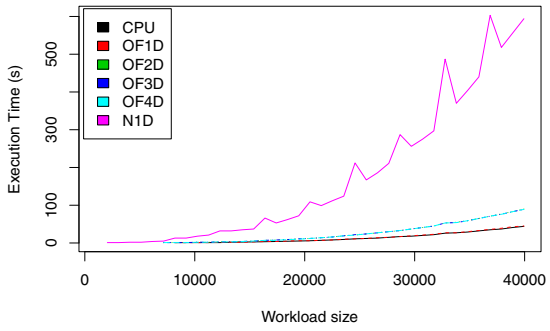

Fig. 7. Execution time (Linpack).

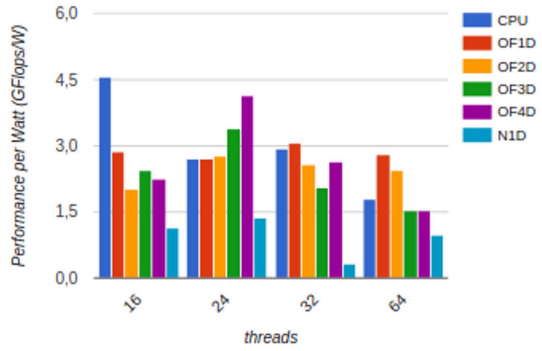

Fig. 8. Performance per Watt according to the number of threads (Linpack).

and 4 GFlops/W with similar variations using 24 threads. But the *host* (CPU) model using 16 threads was the highest value scenario reaching 4.5 GFlops/W. The *host* (CPU) model using 16 threads obtained the best PPW ratio with a value of 4.5 GFlops/W. This figure presents good results compared to the Green500 ranking of June 2017, where only 24 supercomputers are with PPW greater than 4.5 GFlops/W. The *native* model obtained the worst relation with values smaller than 1.5 GFlops/W in all scenarios regardless of the number of threads.

V. Conclusion

The *host* and *offload* models have shown much similarity in most of the simulations performed both in performance and energy consumption, as opposed to the *native* model, which presented lower performance but greater efficiency in energy consumption. The *host* and *offload* models have proven to be more suitable for use with shared memory because processing control always stays in the CPU. The number of cores and threads in most simulations also directly influence performance and energy, where the higher the number of threads the higher the performance. But there are some exceptions. In *host* and *offload* models, scenarios using more than 200 threads have the same performance as using 72 threads (1 thread per CPU *core*). Also, the use of 24 threads demonstrated the worst average performance and higher energy consumption in most of the simulations. Processes with small workloads must consider the use of the *native* model, which does not use any computational resource of the CPU, greatly reducing the energy consumption.

Acknowledgments

We would like to thank Intel Modern Code Partner at Unesp, which provide the infrastructure for the tests, and FAPERGS/CNPq, by the support through PRONEX 12/2014.

References

[1] Shao, Yakun Sophia and Brooks, David: Energy characterization and instruction-level energy model of Intel's Xeon Phi processor, International Symposium on Low Power Electronics and Design (ISLPED) 389–394 IEEE (2013)

[2] Lakomski, Donna and Zong, Ziliang and Jin, Tongdan Optimal Balance between Energy and Performance in Hybrid Computing Applications http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7393697 (2015)

[3] Si, Min and Peña, Antonio J. and Balaji, Pavan and Takagi, Masamichi and Ishikawa, Yutaka MT-MPI: Multithreaded MPI for Many-Core Environments Proceedings of the 28th ACM international conference on Supercomputing - ICS (2014)

[4] Hsu, Chung-Hsing and Kremer, Ulrich The Design, Implementation, and Evaluation of a Compiler Algorithm for CPU Energy Reduction ACM SIGPLAN Notices number 5, pages 38 (2003)

[5] Balladini, Javier and Suppi, Remo and Rexachs, Dolores and Luque, Emilio Impact of parallel programming models and CPUs clock frequency on energy consumption of HPC systems Proceedings of IEEE/ACS International Conference on Computer Systems and Applications, AICCSA pages 16–21 (2011)

[6] Lawson, Gary and Sosonkina, Masha and Shen, Yuzhong Energy Evaluation for Applications with Different Thread Affinities on the Intel Xeon Phi Workshop on Applications for Multi-Core Architectures (WAMCA) pages 54–59 (2014)

[7] Lawson, Gary and Sundriyal, Vaibhav and Sosonkina, Masha and Shen, Yuzhong Runtime Power Limiting of Parallel Applications on Intel Xeon Phi Processors http://ieeexplore.ieee.org/document/7830507/ (2016)

[8] Henao, John Anderson García and Navaux, Philippe Olivier Alexandre and Hernandez, Esteban and Hernandez, Carlos Jaime Barrios enerGyPU and enerGyPhi Monitor for Power Consumption and Performance Evaluation on Nvidia Tesla GPU and Intel Xeon Phi (2016)

[9] Chrysos, George Intel® Xeon Phi™ Coprocessor - the Architecture Hot Chips Conference 2012, Figure 3 pages 1–8,(2012)

[10] Intel Inc Intel ® Xeon Phi Coprocessor x100 Product Family Data Sheet (2015)

[11] Parallel Programming and Optimization with Intel Xeon Phi Coprocessors Introduction Parallel Programming and Optimization with Intel Xeon Phi Coprocessors Handbook on the Development and Optimization of Parallel Applications for Intel Xeon Process http://www.colfax-intl.com/nd/xeonphi/book.aspx, volume 1 (2013)

[12] Green500 - TOP500 Supercomputer Sites https://www.top500.org/green500/ (2017)

[13] Intel Manycore Platform Software Stack (Intel MPSS) — Intel Software https://software.intel.com/en-us/articles/intel-manycore-platform-software-stack-mpss (2014)

[14] 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing Statistical Power and Performance Modeling for Optimizing the Energy Efficiency of Scientific Computing (2010)

[15] Sharma, Sushant and Hsu, Chung-Hsing and Feng, Wu-Chun Making a Case for a Green500 List (2006)

[16] Smejkal Till, Hähnel Marcus, Ilsche Thomas, Roitzsch Michael, Wolfgang E, Härtig Hermann E-Team : Practical Energy Accounting for Multi-Core Systems E-Team : Practical Energy Accounting for Multi-Core Systems (2017)

[17] Rostirolla, Gustavo and Da Rosa Righi, Rodrigo and Rodrigues, Vinicius Facco and Velho, Pedro and Padoin, Edson Luiz GreenHPC: A novel framework to measure energy consumption on HPC applications Sustainable Internet and ICT for Sustainability, SustainIT (2015)

[18] Lorenzo, O G and Pena, T F and Cabaleiro, J C and Pichel, J C and Rivera, F F and Nikolopoulos, D S Power and Energy Implications of the Number of Threads Used on the Intel Xeon Phi (2015)

[19] Netlib HPL HPL - A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers http://www.netlib.org/benchmark/hpl/ (2017)

[20] Intel Xeon Phi Coprocessadores Intel Xeon Phi https://www.intel.com.br/content/www/br/pt/products/processors/xeon-phi/xeon-phi-coprocessors.html (2015)

[21] Giefers, Heiner and Staar, Peter and Bekas, Costas and Hagleitner, Christoph Analyzing the energy-efficiency of sparse matrix multiplication on heterogeneous systems: A comparative study of GPU, Xeon Phi and FPGA 2016 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS) http://ieeexplore.ieee.org/document/7482073/ (2016)