Measurement and Optimization of Energy Consumption in HPC Applications Parallel and Grid Computing

Ivan Tishchenko
Campus Kirchbeg
University of Luxembourg
6 Rue Richard Coudenhove-Kalergi, 1359 Luxembourg
Email: ivan.tishchenko.001@student.uni.lu

Oleksandr Borysov
Campus Kirchbeg
University of Luxembourg
6 Rue Richard Coudenhove-Kalergi, 1359 Luxembourg
Email: oleksandr.borysov.001@student.uni.lu

***Abstract*—Measuring the energy and power consumption of computer systems has always been one of the most important problems on the path towards building the high performance computing systems. The issue of the day in the domain of modern supercomputing is the attempt to deliver the computer systems with the computing performance of exaflop/s.**

The purpose of this work is to present the most interesting techniques in energy consumption measurement frameworks, while describing their advantages and limitations to measure the energy consumption of different types of architectures. Finally, a methodology proposed in this works would allow to select the right power consumption measurement system taking into account the energy efficiency, measurement precision and adequacy of the measurement system.

***Keywords*-High Performance Computing; Energy consumption; optimization;**

1. Introduction

When it comes to the domain of HPC and Grid Computing the power dissipation in data centers increases greatly over time. It has been exhaustively calculated that the cost of maintaining the servers for two full years may be more than the cost of the equipment itself. Fraction of energy spent on the computer systems over the worldwide available electricity is estimated to double in the period of twelve years [1]. In order to decrease the maintenance costs for high performance computing systems, which in return would increase the applicability of the systems. Hardware manufactures introduce improved energy efficiency while delivering slight performance improvements with each iteration of devices.

The main trend which has developed during the recent years is to try to maximize the performance/watt, apart from only maximizing purely performance related parameters. The computing solutions of the future are needed to increase both the energy efficiency and the computation capabilities. The performance of computer systems has met new constraints due to power consumption issues. One common example is the maintenance cost of high performance computing (HPC) systems which in many cases can rapidly exceed the acquisition price of the computer system. Therefore, it is crucial for the new solutions to increase the computing performance (such as flops/sec or others) while maintaining low energy consumption (EC). However, the effort to enhance EC of computing units is not trivial and involves several areas of computing engineering, such as hardware design or software design. [2]

As for the hardware part there is a huge demand for energy resources to run parallel software applications on heterogeneous parallel and grid computing systems. Naturally, the major users of these computing systems are really high demanding, high resource scientific tasks with huge level of parallelism (e.g. weather predictions, plasma physics computations, biology genome computations, large scale network processing etc.) it also usually involves data centers where massive levels of visualization are demanded. [3] Another constrain is that parallel applications usually demand high availability from the system until the completion of a computational job. Obviously because of that reason parallel and grid computer systems generally undergo very high operational and maintenance cost.

The figures from the Energy star's report (Report to Congress on Server and Data Center Energy Efficiency) are a clear indication of the facts of consumption. Based on the figures in 2007 in EPA report in United States of America the data centers alone consumes 61 billion kWh in the year of 2006; which is basically equivalent to the energy needed to power 5.8 million average households. Furthermore the projection is that the numbers in the IT industry will grow exponentially in the following years. In the industry large scale data centers contribute the most to the energy part of the budget. From the figures and facts obviously reducing data center energy consumption is a major scientific challenge in the modern IT for today and for the years to come. According to the report, in 1998, heat loads for dense rack-mount servers were at the level of 5 Kilo Watt per rack but by 2006, however the dynamics increased to 32 Kilo Watt per rack. An increase of almost 7 times is an evident fact of the energy consumption issue, which largely important nowadays [4]

Over the past years hardware manufactures have been reducing the energy consumption of computer systems by designing and manufacturing more efficient hardware. However the resources could be also optimized from the users side by using hardware efficiently. This is why the topic of this work would also aim to draw conclusions regarding the utilization of resources between computational units in order to decrease the energy consumption. For this purpose several computationally intensive HPC applications were used in this work to test on various types of hardware and on various CPU architectures

during the single node and multi node execution. The most useful implication of this work is that it would draw the directions for the optimization of parallel algorithms.

2. Energy consumption data of multi-core CPUs on multiple HPC nodes

Overall, the ultimate purpose of the project is to examine the energy efficiency by measuring the power consumption on various types of multi-core CPU types on various heterogeneous nodes, in different modes (e.g. full lode, idle, partially loading some cores etc.). The hardware side of the project, on which all the power measurements were conducted was provided by the University of Luxembourg, namely by the High Performance Computing group.

All the work was performed on the "Gaia" cluster with the project_pwrmon group's privileges, which gave the ability to conduct a wide range of experiments using various measurement tools. The main choice for the "Gaia" cluster can be explained by the availability of various heterogeneous computer nodes to enable the accusation of data from various CPU types. Moreover the nodes share the same processor architecture that had a positive implication because that the applications compiled on one of the nodes are likely to work adequately on the other ones.

The full list of nodes which was available for the measurements is displayed in the table 1:

Node name	CPU type	Clock rate
gaia-[1-60]	Intel Xeon L5640	2,26 GHz
moonshot1-[1-45]	Intel Xeon E3-1284L v3	1,8 GHz
moonshot2-[1-45]	Intel Xeon E3-1284L v3	1,8 GHz
gaia-[83-122]	Intel Xeon X5670	2,93 GHz
gaia-[123-154]	Intel Xeon X5675	3,07 GHz
gaia-[155-178	Intel Xeon E5-2680 v3	2.5 GHz
gaia-[63-72]	Intel Xeon L5640	2,26 GHz
gaia-[75-79]	Intel Xeon E5-2660	2.2GHz
gaia-[179-182]	Intel Xeon E5-2680 v3	2.5 GHz
gaia-[61-62]	Intel Xeon L5640	2,26 GHz
gaia-[183-184]	Intel Xeon E7-8867 v3	2.5 GHz
gaia-73	Intel Xeon E7-4850	2GHz
gaia-74	Intel Xeon E5-4640	2.4GHz
gaia-80	Intel Xeon E7-8880 v2	2.5 GHz
gaia-81	Intel Xeon E5-4650 v2	2.4 GHz

Table 1: Full nodes list

Due do the restrictions which came from the tools side, not all CPU architectures supported the tools of our choice. Thus we could only measure the power consumptions on certain types of nodes. While selecting the node type for measurements various factors as availability, compatibility with the measurement tools, Clock rates, number of cores were taken into account.

Based on various factors the following nodes have been selected:

- gaia-159
- gaia-81
- moonshot nodes

All of the nodes listed above have the support for the measurement which we have selected.

The hardware of the nodes is the following:

- Dell FC430, with Intel Xeon E5-2680. From the Intel's specification sheets [8] we picked up the parameters which are relevant to the assessment.

Performance	
# of Cores	12
# of Threads	24
Processor Base Frequency	2.5 GHz
Max Turbo Frequency	3.30 GHz
TDP	120 W
VID Voltage Range	0.65V-1.30V
Price	$1745-$1749

- HP ProLiant m710, with Intel Xeon E3-1284L

Performance	
# of Cores	4
# of Threads	8
Processor Base Frequency	1.8 GHz
Max Turbo Frequency	3.2 GHz
TDP	47 W
VID Voltage Range	0.60V-1.35V

- SGI UV2000, with Intel Xeon E5-4650

Performance	
# of Cores	10
# of Threads	20
Processor Base Frequency	2.4 GHz
Max Turbo Frequency	2.9 GHz
TDP	95 W
VID Voltage Range	0.65V-1.30V
Price	$3616-$3620

In order to properly perform the power measurements on a single node and multi nodes we wrote the scripts multiNodeTestScript and singleNodeTestScript for the automated booking of the nodes and execution of energy measurements tests, before conducting any experiments. We used the scripts numerous times during the project and which saved us a lot of time. (both scripts are available in the submission archive)

3. Power measurement tools

Before diving into the details of power consumption, one would have to select the adequate measuring tools to acquire appropriate results regarding the power consumption which would later enable to draw the conclusions and do statistical analysis on the data, this is what this section is concerned about, including the subsections with the specific tools. This section of our work will describe various methodologies and tools to measure power and energy of the CPU. The devices presented in the last section would be measured with the tools, in addition

there will be provided an overview of the most state of the art tools used nowadays and the description of their advantages and disadvantages with regard to each other and in general.

The variety of tools which was originally offered for the project is the following:

- likwid
- Intel Performance Counter Monitor
- libEC
- Kwapi

The detailed overview of tools used in the project would be described in the succeeding subsections.

3.1. Lkwid-powermeter tool

The first choice for our project of energy measurement tools is Likwid:
"LIKWID (“Like I Knew What I’m Doing”) is a set of easy to use command line tools that addresses several key problems encountered at performance-oriented programming in a Linux environment." – [10].

It is a Linux command line tools, which is very easy to use, very portable, efficient and extensible. Moreover, it is an open source project. However, it gives to its users the access to some important measuring and managing functionalities with the help of which we could access privileged sections of the operating system and even hardware registers. These were main advantages of the tool over the other ones.

3.2. Intel Performance counter Monitor (PCM)

PCM is a standalone cross platform command-line application for measuring the power consumption. It works is a turbostat fashion (periodically prints the measurement over in interval of time). The time frame is specified with a parameter. PCM is quite a vast and complex measuring tool, however in our case a utility target towards power measurement hold the most interest (**PCM Power**).

With the utility one could measure the core performance of the Xeon E5 or E7 processor types.

3.3. libec-powermeter tool

The main purpose of the library libec is to help developers to create a new modular libraries for power estimation. The library was implemented in C++ which allows easy to extend and maintain it. Also it was implemented under the GNU General Public License (GPL) version 3.0 or later. It can be downloaded from here [4]. This library contains a set of sensors as input variables in several models of capacity. Information provided by the sensors comes mainly from API kernel Linux, / SYS and / file system Proc. However, these sensors can be extended for collecting various data coming from any particular source data.

The libec sensor can be implemented at two levels: machine and/or applications. The applications level containing all sensors that can be directly associated with a process indeficator(PID), these sensors are mainly related to software use such as performance counters. At the same time, the level of the machine not only the total value for all processes, and measure some physical properties which can not be associated with PID, such as CPU heat.

3.4. Choice of measurement tools

This section will regard the choice of measurement tools. We have decided to choose likwid as our main power consumption measurement tool. One of the primary reasons for giving for the selection of this tool is the possibility to wrap applications around and measure their power consumption parameters. One of our various tasks entailed the measurement of power consumption of an execution on various multiply nodes. For this specific purpose we needed to use technology for distributed computing (in our project we chose MPI) to run an application on different nodes. The tool name "Likwid" gives a programmer to wrap around such an execution of an application and get the measurement data. However, during the course of this stage we stumbled over some a issue the detailed description of which would be covered in great details.

Note: It is a fact that Likwid allows to run on nodes and measure energy consumption, however unfortunately we can not adjust the number of processes primary due to the reason that likwid acts as wrapper of a processor. That is why the only solution is to develop a program which puts a load on a CPU by creating threads. And further runs the examples of a process on each host.

4. The test collection for measuring CPU’s energy consumption

It is a recognized fact that we all know that program speed and energy consumption of a CPU is highly correlated to the cost of application’s math performance. For this reason we made a decision to develop and design our tests based on various mathematical operations. A collection of these tests would allow us to gain an understanding of the costs of different mathematical operations on a CPU as well as it would reveal the numbers regarding power consumption of different applications. With this knowledge in mind we will be able to produce real HPC-programmers.

Our tests collection is divided into four main categories: Green, Yellow, Red and Performance drops. In additional to the previous ones we will implement a special test which would allow to check if it is possible to optimize the program by using a special technique. The purpose of this test will be described above in section Optimization of energy consumption. In addition to that, we decided to set up the duration of 60 seconds for each individual test and 10 seconds of break between tests respectively.

4.1. Green tests

These category of tests includes three types of sub-tests which divided by execution hardness for a CPU. As the names suggests us the green operations are the cheapest for the CPU. The sub-tests are the following ones:

- addition, subtraction, comparison (1)
- abs (2)
- multiplication (4)

4.2. Yellow tests

Next category of tests of our choice includes two sub-tests: division and modulus operations. These tests are know to be more than twice as expensive as multiplication (a weight 10). The sub-categories are:

- division
- modulus

4.3. Red tests

The last category of tests of mathematical related tests is rather heavy on the CPU compared to the previous ones. Moreover one of these tests will show very interesting result which we will cover in great details later in the results sections. The sub-categories are:

- exp (50)
- sqrt (55)
- sin (60)
- asin (80)
- pow (100)

4.4. Performance drops test

The last category is Performance drops test. This kind of test allows to measure CPU's power consumption during the change of a processor's state from idling to active state. Also it would allow to make a discovery how fast a CPU reaches it's maximal performance from idling mode.

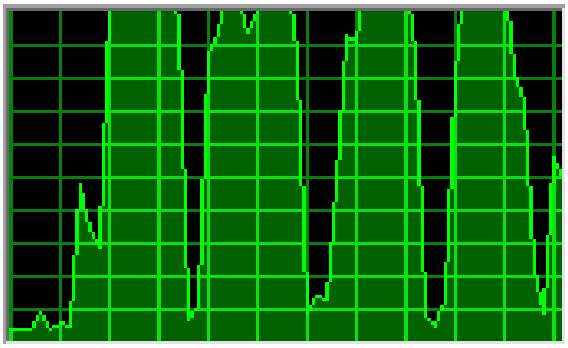

Figure 1: Demonstration of applications' performance drops

5. moonshot vs gaia-157

For the first test we have decided to take two types of nodes namely, moonshot1-1 and gaia-157. From the specifications described earlier, we know that Moonshot nodes have one CPU, which is Intel Xeon E3-1284L v3 on the other hand gaia-157 node has two CPUs of type Intel Xeon E5-2680 v3. We will make a comparison of power consumption of one CPU from each nodes. The representation is in table 2.

5.1. Common energy consumption results

This section would describe and demonstrate the results of energy consumption, which was measured by means of conducting a wide variety of tests described earlier. The plot below described an energy consumption of one CPU (PWR1) per one core with the step set to 0.5 seconds. To collect this information we used likwid-perfctr command-line utility with a group ENERGY.

	E5-2680 v3	E3-1284L v3
Cores	12	4
Threads	24	8
Clock speed	2.5 GHz	1.8 GHz
Turbo clock	3.3 GHz	3.2 GHz
Performance	12.03 pt/W	5.57 pt/W
CPUs in SMP	2	1
Consumption	97.5W	38.19W

Table 2: Comparison of CPUs [6]

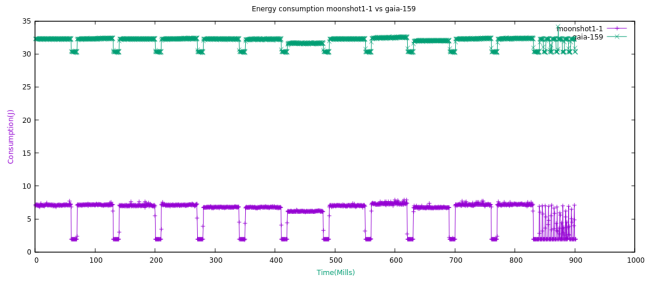

Figure 2: Energy consumption during tests

The energy consumption of gaia-159 is significantly higher than the one obtained on the moonshot1-1 node. Also one could clearly see that during the test number 7 (SQRT test), a processor uses the least amount of energy than during the other tests.(Each test separated by 10 second pit). The conclusion about it will followed up later in the conclusion sub chapter.

5.2. Application calculation drops

This plot is based on the information about energy consumptions by CPUs per one core during the so called "Drop" test. As it is clearly seen from the visual part the

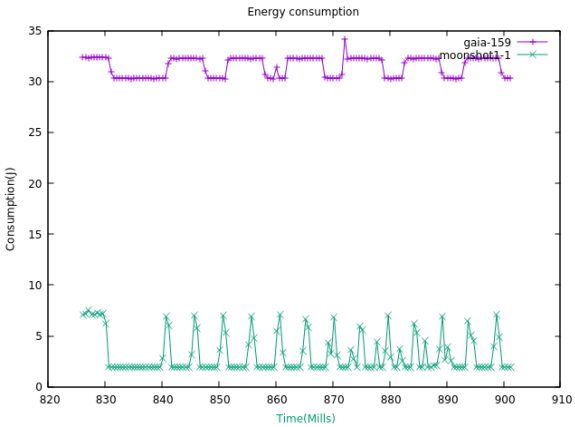

Figure 3: Energy consumption tests during calculation drops

gaia-159 node acts as a theoretical prediction.(Sequence

of calculation and four seconds pit). However moonshot node acts in the another way: a very quick calculation than four seconds of break etc.

This situation could be possible in only one way: moonshot activates its Turbo Boost mode during the calculations, because the amount of calculation for both processors was exactly identical.

Note: Intel Turbo Boost technology is a technique implemented by the Intel corporation in certain versions of its processors that enables the processor to run above its base operating frequency via dynamic control of the processor's clock rate. Turbo Boost is activated when the operating system requests the highest performance state of the processor.

5.3. Consumption of energy during each test

The following plot would demonstrate the energy consumption by all the CPU consumers per one core.

The plot demonstrates somewhat inordinate behaviour of energy consumption namely during the test number 8 (Sinus test). However further measurement demonstrated and proved that this is a common issue which sometimes has a tendency to occur during measurements because of various factors which are not in out control.

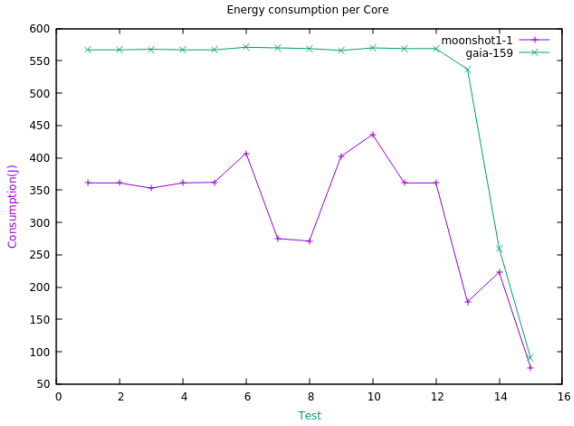

Figure 4: Common energy consumption of tests

We can see that moonshot has jumps of energy consumption while gaia-159 has stable consumption for high performance parts. Further mention is that before, moonshot has low consumption during test number 7(SQRT test) the key for this mastery could be a built-in component which could be used by moonshot's CPU during the calculations of SQRT operation.

5.4. Conclusions

Our tests show that moonshots have somewhat very interesting low cost processors which still can show pretty unexpected results. We think that the main difference between gaia-[155-178] nodes and moonshot1,2-[1-45] CPUs are their power controllers. The tests illustrate that the moonshots CPU's energy controller try maximal reduced energy consumption, while gaia[155-178] CPU's energy controller mostly care about stable energy supply for the processor. This is the main difference between thees CPU types.

6. moonshots vs gaia-157's CPUs

In this section we compare energy consumption of an entire node gaia-159 (Two CPUs) and two moonshot nodes (moonshot1-26 and moonshot1-32) which act as a machine.

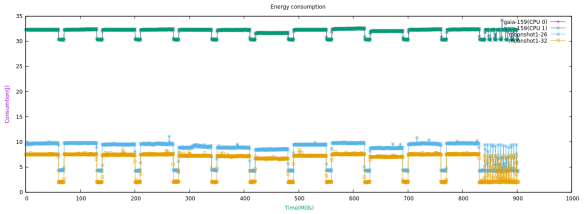

Figure 5: Energy consumption during tests

The plot shows us that the power consumption of processors of gaia-159 and moonshot1-26,32 are different. Namely that they have the same consumption during tests but fluctuating based on the same constant.

6.1. Application calculation drops

The drops of energy consumption appears to be almost the same as for the single CPU tests fluctuating based on a constant.

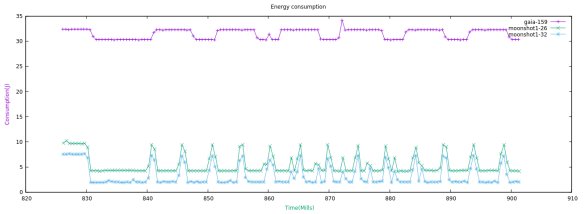

Figure 6: Energy consumption tests during calculation drops

6.2. Consumption of energy during each test

This plot spots a mistake of measurements which we mentioned above and demonstrates a difference of CPUs.

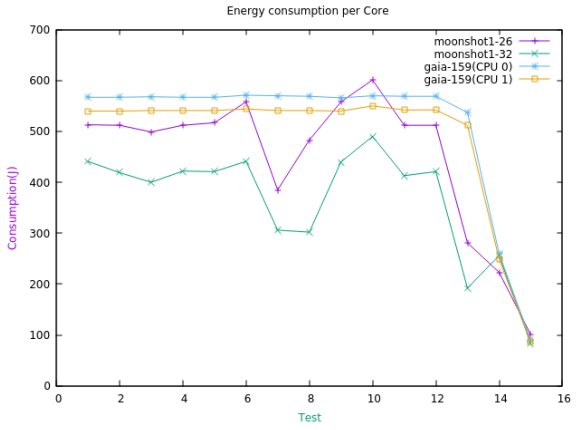

Figure 7: Common energy consumption of tests

6.3. Conclusions

The measurements show that CPUs act during the test in almost the same manner as in the single mode. But we can see the difference in energy consumptions during measurements by some stable coefficient. We think this happens because of different exploitation levels of the processors.

7. gaia-159 vs gaia-81

The last comparison will be drawn between gaia-159 and gaia-81. Gaia-81 has sixteen CPUs Intel Xeon E5-4650 v2. We will compare power consumption between one CPU from each nodes.

	E5-2680 v3	E5-4650 v2
Cores	12	10
Threads	24	20
Clock speed	2.5 GHz	2.4 GHz
Turbo clock	3.3 GHz	2.9 GHz
Performance	12.03 pt/W	5.57 pt/W
CPUs in SMP	2	4
Consumption	97.5W	77.19W

Table 3: Comparison of CPUs [8]

7.1. Common energy consumption results

Tests shows that energy consumption of gaia-81 is mostly the same for all the tests. Also gaia-159 energy consumption is significantly higher than gaia-81.

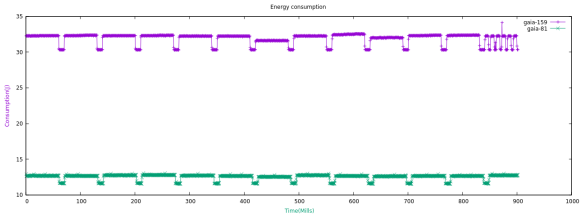

Figure 8: Energy consumption during tests

7.2. Application drops

This plot holds the most interest because it demonstrates that gaia-81's energy controller does not reduce the power consumption even if a processor is in idle mode.

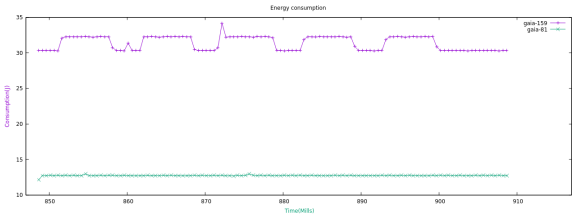

Figure 9: Energy consumption tests during calculation drops

7.3. Consumption of energy during each test

We can see that gaia-81 energy consumption is more slight than gaia-159 it means that our assumption is potentially correct and a power controller of gaia-81 worse than the respective value of gaia-159.

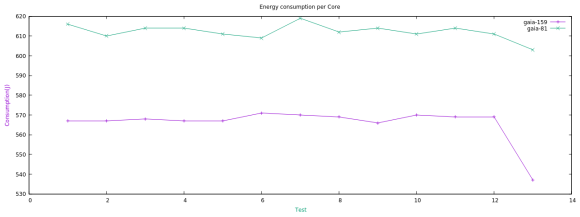

Figure 10: Common energy consumption of tests

7.4. Conclusions

During this test we established that gaia-81 CPUs has very inflexible power energy controller. We assumed that this is the cost which paid such type of CPU for 10 physical cores in one cheap.

8. Optimization of energy consumption

There are various ways to optimize the software side. This section would describe such the ways.

One of the common issues is branch predication [18] fail. For the purposes of explanation of branch predication we would consider an IF statement. On the assembly language level a conditional statement is a branch instruction (e.g. JMP in x86 assembly). One could think that a processor halts and waits for the previous instruction to complete.

However, this naive assumption is wrong. Modern CPUs have long pipelines and it takes a long time to "start" or "stop". What is done is the prediction approach. Thus there are 2 logical outcomes: continue the execution because the guess was right or flush the pipeline and roll back the branch and just restart to the opposite path. Overall, if the processor guesses right every time, the execution will never have to stop, however if the processor guesses wrong too often, we would loose time while rolling back, restarting and flushing out pipelines.

How could one exploit the branch prediction described at the previous paragraph to boost the performance and lower the power consumption The task of a branch predictor is to identify a pattern and follow it. To utilize branch predictors work one has to aim to have well-behaved branches in the software part. A well behaved patter will consecutively go to the same direction many times thus it would allow branch predictor to correctly predict the branch. This way even a simple counter will correctly predict the branch.

We made a decision to include in our testing package two tests (The search in an unsorted array (test number 14) and the search in a sorted array (test number 15)) in order to verify branch prediction. The results appeared to be quite incredible. On gaia-159 we achieved 2,77 times lower energy consumption just by sorting an array before start search. This is just a minor demonstration of this powerful optimization technique.

The examples are the following ones:

Listing 1: NotOptimized.cpp

```
void notOptimized() {

        // Generate data
        const unsigned arraySize = 32768;
        int data[arraySize];

        for (unsigned c = 0; c < arraySize; ++c)
                data[c] = rand() % 256;


        volatile long long sum = 0;

        for (unsigned i = 0; i < 100000; ++i)
        {
                // Primary loop
                for (unsigned c = 0; c < arraySize; ++c)
                {
                        if (data[c] >= 128)
                                sum += data[c];
                }
        }
}
```

Listing 2: optimized.cpp

```
void optimized() {

        // Generate data
        const unsigned arraySize = 32768;
        int data[arraySize];

        for (unsigned c = 0; c < arraySize; ++c)
                data[c] = rand() % 256;

        // !!! With this, the next loop runs faster
        sort(data, data + arraySize);

        volatile long long sum = 0;

        for (unsigned i = 0; i < 100000; ++i)
        {
                // Primary loop
                for (unsigned c = 0; c < arraySize; ++c)
                {
                        if (data[c] >= 128)
                                sum += data[c];
                }
        }
}
```

Single lines of code can sometimes significantly decrease energy consumption and increase performance.

8.1. Optimization by replacing mathematical operations

Another way of an optimization is a technique of replacing of some mathematical operations by another ones. For example if one can multiply integers with 2 then it is easier and some very cheap bit-shifts can be used. Example 6 = 110, 6 « 1 = 1100 = 12.

If one can replace division by multiplication you do get a speed-up of more than two. A side-effect is that errors (especially floats) multiply too and you can end up with less precision.

9. Problems during project implementation and their solutions

This section would regard the problems which obvious to occur over the course of this type of a a project we would also reveal our solutions to the these problems.

All the problems can be categorized in the following categories:

- Cluster related issues.
- Software side related issues.
- Hardware side related issues.
- Measurement tools related issues.

Over the project went well which resulted in adequate measurements. However, some of the issues which occurred were: high load of the cluster which resulted in long waiting times to book a full node and even longer waiting times for the multiple nodes, to solve this issue we used scripting to automate the compilation, execution, measurement and creation of log data (Please see the attached files for more details).

The CPU architectures such as gaia-[1-60], gaia-[61-62], gaia-[63-72], gaia-73 were not able to support the measurement tools proposed by the project, this way we had to limit the choice to certain type of architectures which support the measurement tool, before the start of the actual data acquisition we checked the comparability of each processor type with the tool of our choice.

Another issue was to extract the meaningful data from the massive log files produced by measurement tool likwid-perfctr, the solution were to look source files to understand output and to plot the data with the open source plotting library in our case GNU Plot [17] and use statistical analysis to extract the adequate data.

10. Conclusions

The measurement of the energy consumption is a very open field which is rich for the discoveries for researches and scientists. In this work we described, analyzed and compared various types of energy efficiency metrics tools on a wide range of CPU types and micro architectures. We have described the ideas and concepts to discover where the energy consumption comes from in the domain parallel applications in HPC.

By the results of conduction of various measurements and plotting which are available in this work, one could obtain a more profound understanding how software implementation and various micro architecture types are correlated to each other and how they affect the overall power and energy consumption.

The next step of this work would be to identify the patterns from the computational data and find a technique to optimize the performance making it more energy efficient.

We discover the information about the Turbo boost in larger details. Overall, it accelerates processors and graphics performance for peak loads, automatically allows processor's cores to run faster than the rated operating frequency. We had an opportunity to see this technology in action and now we know how it can be useful in our future projects in various domains.

We verified a very interesting and useful concept of "Branch Prediction" and made tests of it on five various node types. All of the nodes have demonstrated great results with this type of an optimization. This gives a proofs the statement that even some very minor changes in code (in our example it was as little as one line of the application's code) can largely impact the overall application's performance, which in return will decrease the overall power and energy consumption.

Obviously the topic of this work is not optimization thus we did not cover much of this field of measurement and optimizations. However our next step would be to

discover some optimization techniques as we got to obtain some great essential knowledge about the concepts of it.

Acknowledgments

The authors would like to thank the entire academic staff who was in charge of the "Parallel and GRID Programming" course which is a part of the "Master of Information and Computer Science" program at the University of Luxembourg for the academic content which the result of which is this work. The authors would also like to thank the administrative section of the High Performance computing unit at the University of Luxembourg who provided the hardware infrastructure by giving an access to the Gaia cluster and all of it power measurement tools which made the project possible.

References

[1] Vereecken, W., Van Heddeghem, W., Colle, D., Pickavet, M., Demeester, P., *Overall ICT footprint and green communication technologies*, 4th International Symposium on Communications, Control and Signal Processing (ISCCSP), pp. 1-6, March 2010

[2] G. Calandrini,1, A. Gardel, I. Bravo, P. Revenga, J. Lázaro, and F. Toledo-Moreo, *Power Measurement Methods for Energy Efficient Applications*, June 2013.

[3] D.B Srinivas, Puneeth R.P, Rajan M.A, Sanjay, *An integrated framework to measure the energy consumption of a parallel application on a computational grid* , IJCSNS International Journal of Computer Science and Network Security, VOL.16 No.7, July 2016.

[4] Energy star power consumption report, *Report to Congress on Server and Data Center Energy Efficiency* , energystar.gov/ia/prod/EPA_Datacenter/_Report/_Congress_Final1.pdf

[5] Cupertino, L.F., *Energy consumption tools web-page (www.irit.fr/∼Leandro. Fontoura-Cupertino/ectools/)*, April 2013

[6] *CPU compare E5-2680 v3 vs E3-1284L v3 (http://cpuboss.com/cpus/Intel-Xeon-E5-2680-v3-vs-Intel-Xeon-E3-1284L-v3)*

[7] High Performance Computing in Luxembourg, The Gaia Cluster, *Computing Capacity (https://hpc.uni.lu/systems/gaia/)*

[8] Your Source for Intel® Product Specifications, Intel® corporation, *Server Processors specifications (http://ark.intel.com/products/)*

[9] Robert Schöne, Jan Treibig , Manuel F. Dolz, Carla Guillen, Carmen Navarrete, Michael Knobloch e and Barry Rountree f, *Tools and methods for measuring and tuning the energy efficiency of HPC systems* , Scientific Programming 22 (2014) 273–283 273, DOI 10.3233/SPR-140393 IOS Press, 2014.

[10] J. Treibig, G. Hager, and G. Wellein, *"Likwid: A lightweight performance-oriented tool suite for x86 multicore environments*, in 2010 39th International Conference on Parallel Processing Workshops, pp. 207–216, IEEE, 2010.

[11] T. Willhalm, *Intel performance counter monitor - a better way to measure cpu utilization (https://software.intel.com/en-us/articles/intel-performance-counter-monitor)*, August 2012.

[12] L. F. Cupertino, G. Da Costa, A. Sayah, and J.-M. Pierson, *Energy consumption library*, in Energy Efficiency in Large Scale Distributed Systems, pp. 51–57, Springer, 2013.

[13] F. Rossigneux, J.-P. Gelas, L. Lefevre, and M. D. de Assun¸cao, *A generic and extensible framework for monitoring energy consumption of openstack clouds*, arXiv preprint arXiv:1408.6328, 2014.

[14] B. L., *Intel developer zone: Measuring application power consumption on the linux operating system (https://software.intel.com/en-us/blogs/2013/06/18/measuring-application-power-consumption-on-linux-operating-system)*, arXiv preprint arXiv:1408.6328, October 2013.

[15] *CPU compare E5-2680 v3 vs E5-4650 v2 (http://cpuboss.com/cpus/Intel-Xeon-E5-4650-v2-vs-Intel-Xeon-E5-2680-v3)*

[16] Gough, Corey, Steiner, Ian, Saunders, Winston, *Energy Efficient Servers. Blueprints for Data Center Optimization*, First Edition DOI 10.1007/978-1-4302-6638-9 Apress, 2015.

[17] GNU Plot *Open source library for plotting the data (http://www.gnuplot.info/)*

[18] Branch prediction *https://en.wikipedia.org/wiki/Branch_predictor)*

[19] Branch prediction explanations *http://stackoverflow.com/questions/11227809/why-is-it-faster-to-process-a-sorted-array-than-an-unsorted-arrayr)*

[20] L. Adhianto, S. Banerjee, M. Fagan, M. Krentel, G. Marin, J. Mellor-Crummey and N.R. Tallent, *Hpc toolkit for performance analysis of optimized parallel programs*, Concurrency and Computation: Practice and Experience, pp. 685–701 2010.

[21] B. Goel, S.A. McKee, R. Gioiosa, K. Singh, M. Bhadauria and M. Cesati, *Portable, scalable, per-core power estimation for intelligent resource management*, in: IGCC, pp. 135–146 2010.

[22] J.I. Aliaga, M. Bollhöfer, A.F. Martín and E.S. Quintana-Ortí, *Exploiting thread-level parallelism in the iterative solution of sparse linear systems*, in Parallel Computing 37(3). 183–202 2011.

[23] M.E.M. Diouri, M.F. Dolz, O. Glück, L. Lef‘evre, P. Alonso, S. Catalán, R. Mayo and E.S. Quintana-Ortí, *Solving some mysteries in power monitoring of servers: Take care of your wattmeters!*, Energy Efficiency in Large Scale Distributed Systems, J.-M. Pierson, G. Da Costa and L. Dittmann, eds, Lecture Notes in Computer Science, Springer, Berlin/Heidelberg 3–18. 2013.

[24] Z. Zhou, Z. Lan, W. Tang and N. Desai, *Reducing energy costs for IBM blue gene/p via power-aware job scheduling*, in: Workshops on Job Scheduling Strategies for Parallel Processing (JSSPP), 2013.