IMTeract Tool for Monitoring and Profiling HPC systems and applications

Violeta Holmes, Christopher Newman, Matthew Newall, Daniel Munnings
School of Computing and Engineering
The University of Huddersfield
Huddersfield, United Kingdom
{v.holmes, c.newman, m.newall, d.munnings}@hud.ac.uk
Gillian Arnold
Tectre Ltd
Huddersfield, United Kingdom
gillian.arnold@tectre.com

***Abstract*— Energy usage of computing equipment is an important consideration and energy inefficiency of computer systems is identified as the single biggest obstacle to advances in computing. Research into low-energy computing products ranges from operating system codes, applications and energy-aware schedulers to cooling systems for data centres. To monitor energy consumption in data and HPC centres it is necessary to develop tools for measuring the energy usage of computer equipment and applications. We have developed power measuring apparatus and a tool, IMTeract, for measuring energy consumption of HPC applications. IMTeract was used for energy usage profiling of HPC clusters running FLUENT and DL-POLY software and a GPU cluster running different implementations of an FFT algorithm. Our experimental results are encouraging and suggest that the IMTeract tool can be used to measure the CPU, Memory, Disk I/O and Network I/O for an application or a process and report on the energy used.**

Keywords- Energy Efficient Computing, Energy Utilisation Profiling, DCIM tool, HPC systems and applications

I. INTRODUCTION

The energy consumption of computing systems is becoming a major area of investigation, in an effort to design more energy-efficient hardware and software.

It is estimated that ICT consumes 9% of electricity and Data Centres (DC) 1-2%. The computing power of the top 10 High-Performance (HPC) systems has increased on average by a factor of 1.9 every year during the last 6 years. The power consumption of the leading-edge supercomputers has reached a level of more than 10 MegaWatts (MW) and continues to grow [1]. Accurate measurement of energy usage is a prerequisite for its reduction.

Data centre managers need to make cost-based efficiency changes to their DC equipment and applications. They lack heterogeneous and real time figures on the cost of power consumption and cooling by application, particularly where there is a heterogeneous environment which includes older equipment and consequently cannot easily understand and relay the cost vs. value of compute for their organisation.

The trade organization Digital Europe (formerly EICTA) has committed to reduce European ICT-related carbon emissions by 20% by 2020 [2].

In 2013, Innovate UK (formerly Technology Strategy Board) has funded a number of Energy-Efficient Computing feasibility studies to encourage technologies which can reduce the energy burden of computing systems [3]. One such project was the IMTeract project run jointly by Tectre Ltd. and the University of Huddersfield UK. The aim of this project was to develop a tool – IMTeract which would measure energy used by server, storage and networking infrastructure, and application data.

Similar tools do exist but are proprietary (Raritan) [4], owned by the vendor selling the hardware, or by virtualisation software vendors. Also, they estimate energy costs only for one part of the infrastructure and not all simultaneously. Furthermore, the vendor benchmarks for energy efficiency are highly tuned and the reality might be somewhat different.

In this paper we present the results of the IMTeract project and developed tool (prototype). This tool should be able to connect to the equipment sited within a data centre, such as standard servers and storage and dedicated HPC systems, and measure unobtrusively the energy consumption of an application or process. The tool should measure the CPU, Memory and Network I/O for an application or process and report on the energy used by the process. In addition, the tool should be capable of benchmarking applications, which can be used to predict when the resources will reach their maximum and facilitate better utilisation of the equipment for the same energy consumption.

II. BACKGROUND

Improving the energy efficiency of Data Centres has been an attractive research topic for both academia and industry. A few key drivers for energy efficiency in the data centre have emerged over the last 5 years in the UK:

- Demand to reduce cost across the business, with electricity expense being major concern,

- Pressure to limit new infrastructure investments such as building new out-of-town Data Centres to house even more computing facilities,
- Regulation such as Carbon Reduction Commitment (CRC), the EU Code of Conduct for Data Centres and the Energy Saving Opportunity Scheme (ESOS) regulations [15].
- Information Technology, Data Centres, Key Performance indicators – Power Usage Effectiveness (PUE) and the pressure to achieve 100% efficiency.

Data Centre Infrastructure Management (DCMI) tools are needed to monitor, measure, manage and/or control data centre utilisation and energy consumption of all IT-related equipment and infrastructure components such as power distribution units and air conditioning equipment [16].

DCMI vendors have been creating tools for monitoring energy usage of latest equipment. Current tools and the DCIM implementations cannot automatically acquire data from older segments of the hardware estate and provide a view of power consumption by application.

In terms of data centres infrastructure, the well-known Power Usage Effectiveness (PUE) metric has help the providers assess and improve the energy efficiency. However, analysing software's energy consumption is also considered an important requirement for profiling energy usage optimisation.

There are a number of existing monitoring tools that should be considered in this context.

A. System monitoring tools

Ganglia [17] is an open-source performance and configuration tool that collects data from a daemon on each OS and then creates web server-based graphical tools to draw performance data and show the configuration. It has been ported to an extensive set of operating systems and processor architectures. It needs to be installed on the servers, and as such is not suitable as a portable tool that can be used on any data centre infrastructure.

Whilst Ganglia is aimed at monitoring a number of servers working on the same task to achieve a common goal - such as a cluster of web servers, Nagios [18] is aimed at monitoring servers, services on servers, switches, network, etc. and will send alerts based on set criteria.

Ganglia and **Nagios** are used for node health monitoring mainly in high performance computing (HPC) environments, but they could be used in clouds and hosting centers.

Intelligent Platform Management Interface (IPMI) from Intel [19] defines a set of common interfaces to a computer system which can be used to monitor system health. IPMI consists of a main controller - the Baseboard Management Controller (BMC) and other management controllers distributed among different system modules - Satellite Controllers (SC). Amongst other pieces of information, IPMI maintains a Sensor Data Records (SDR) repository which provides the readings from individual sensors present on the system, including, sensors for voltage, temperature and fan speed.

The **PowerPack** [20] framework is a set of toolkits composed of hardware and software component; sensors, meters, circuits and data acquisition devices from National Instruments – NI Labview, that enable direct (intrusive) power measurement and instrumentation. The hardware and software enable component-level power measurement, and synchronization between power profiles and application code.

Machine Guided Energy Efficient Compiler framework - **MAGEEC** [21] is an open source project which combines work on compilation options which save energy with work on machine learning, to create a compiler framework that is capable of generating code that has improved energy efficiency.

MAGEEC Researchers from Bristol University, UK, have created an energy measurement board which can be applied to a range of embedded architectures with focus on providing physical energy measurement techniques as opposed to mere mathematical models of energy consumption. The creation of a set of benchmarks (the so-called Bristol/Embecosm Embedded Benchmark Suite, or BEEBS) for comparing runtime and energy performance of programs on embedded architectures is an integral part of MAGEEC tool which is still in development and not yet evaluated in HPC environments.

B. Software Performance Analysis Tools

A variety of automated performance analysis techniques have been developed for profiling complex computer applications. Some of these analysis tools implement simple static techniques, and others rely on advanced dynamic mechanisms to obtain application statistics.

The static analysis tools do not modify the binary image of an application, and rely on source code instrumentation or sampling to obtain results [22]. When recorded, results can be analysed to identify bottlenecks in a program.

The use of static analysis tools can cause unintended side effects, since they are inserted into a set of codes (Compile-time Instrumentation Tools–CIT), or require external sampling routines for data collection (Sampling Tools – ST). This can result in system slowdown because of the overheads introduced by statistics gathering, and can significantly impact the performance of applications.

Hardware counting tools (HCTs) use on-chip programmable event counters to gather information about the state of the processor, and support analysis of applications at execution level. The HCTs are configured to monitor the application execution events. The applications are paused at certain intervals for statistics gathering, similar to the Sampling Tools.

An example of a static tool is **Intel vTune** [23]. vTune is a cross-platform performance and combines the functionality of both an HCT and ST into a single compound tool. It is optimised for use with Intel's own processors and it is incompatible with non-Intel devices.

Dynamic analysis tools rely on binary-level alterations to gather statistical data from an application and can be classified as binary instrumentation or probing.

The Binary Instrumentation Tools (BIT) can inject analysis routines into any locations within an application binary, and record performance data. The Probing tools use the routines embedded in the shared libraries and the kernel to obtain information. These tools are intrusive and modify the structure of the applications they profile, hence the programs could run slower while being analysed due to the increased overhead.

Pin [24] is one of the BIT tools and it provides a Linux-based software development framework for defining portable dynamic instrumentation routines. Pin utilises the Ptrace debug interface provided by the Linux operating system to gain control of an application that is executing on the system and inject the Pin executable into it to gather analysis data.

Julemeter [25] is defined as a software tool that estimates the power consumption of a computer. It estimates the power usage of individual components (CPU/monitor/disk) while the tool is running. However, Joulemeter only works on Windows.

All the above tools require installation on the system under observation, and are intrusive technologies since they might require modification of the existing systems.

In order to measure power, the MAGEEC and PowerPack systems use in-line sensor devices; hence they are intrusive tools at a hardware level.

Intel vTune and Pin are intrusive software tools. They affect the performance of the applications due to the increased overhead while being analysed. In addition, software analysis and/or instrumentation routines created with one tool are typically incompatible with all other implementations, and are tied to a specific operating system.

In order to overcome the shortcomings of the system and software analysis tools outlined above, we have designed and implemented the IMTeract tool.

The IMTeract tool was designed to be unobtrusive in hardware and software/application energy profiling. This tool can profile both Linux and Windows systems, and a variety of CPU and GPU based systems. It is able to profile both old and new data centre equipment.

III. IMTERACT TOOL

IMTeract is implemented as a web application running on an energy-efficient Windows server [5]. It uses Simple Network Management Protocol (SNMP) to acquire data from a system under test (SUT), gather detailed information on the application, middleware and operating systems activity – workload (WUT), and store all this data in a database (DB) as shown in Figure 1.

The tool consists of a number of non-invasive AC Electrical Current Monitoring devices (clamps) connected to a rack monitoring unit, and a low-power server running PHP scripts. Because it uses non-invasive AC monitoring devices and network-attached devices it has the unique feature of being an unobtrusive power measurement tool. It captures power usage data and performance data in a series of scans. A rack monitoring unit [6] is used to capture power usage data in real-time. The workload data (workload profile) is expressed in values of Watts drawn, bytes of memory used, bytes transferred, etc. rather than as percentage of utilisation. The IMTeract captures the information about the system under test - memory size, processor type, frequency and Power Supply Unit as seen in Figure 2.

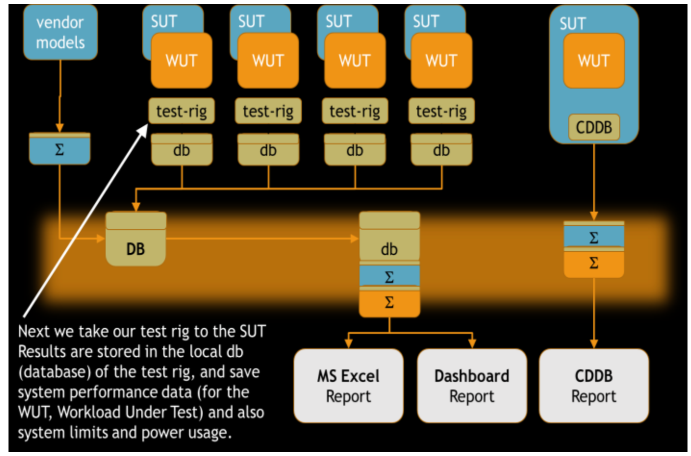


Figure 1. IMTeract Tool.

Captured data is used to generate three types of reports: Survey tasks and dates, comparison of Surveys and a predictive model. Hence, the tool can report the results of the data collection surveys and the comparisons between the same system performance under different workloads, or a vendor's predictive model as shown in Figure 1.

The data is organised in tables containing the individual performance data for each process (db), system level combined performance data for all processes (CDDB), and the power usage and network I/O for the entire system. The data can be visualized using its analysis and reporting system.

A. *The System Survey*

The system survey normally runs a number of scans every two minutes using a Simple Network Management Protocol (SNMP), a popular protocol for network management. It is used for collecting information from, and configuring, network devices such as servers, switches, and routers on an Internet Protocol (IP) network. It was chosen because it is platform-neutral, and is often already installed on the target UNIX and Windows systems. SNMP is designed to be deployed on the largest possible number of network devices, to have minimal impact on the managed nodes, and to have minimal transport requirements. SNMP agents expose management data on the managed systems as variables. For example the SNMP variable for memory usage 'appRAMUsed_Kb' shows how much memory was in use by a process at the time of scanning. These values are summed for all concurrent processes to determine the total system memory requirements. The CPU utilisation and the data for network usage can be also collected using SNMP.

Most of the power benchmarks such as SPECPower give average power usage for the entire system (e.g. server). However in order to measure power at a process level we have used a rack monitoring device that can be interrogated via SNMP through PHP scripts. This device has an array of switches and relay ports for sensors to be connected; it enables non-intrusive power measurement.

In order to evaluate IMTeract tool we have conducted a number of surveys on two production clusters SOL and VEGA in the Data Centre 2 at the University of Huddersfild [7].

IV. IMTERACT TOOL EVALUATION ON SOL

The SOL Node – SPARC server, is one of the worker nodes in the SOL cluster. Each node in the SOL cluster was already configured for SNMP, with *snmpd* deamon enabled and running. The current draw was measured using rack monitoring device and one clamp on the power cabling to the SPARC server. The monitoring comprised a number of scans at 120-second intervals. The data generated from each session was written to several tables in a SQL database. The IMTeract tool uses SNPM to interrogate specific IP addresses and collects data about the hardware, CPU usage, memory usage, HD usage, network interface card and throughput. The resulting data can be used to determine the energy efficiency of the device. The workload was a 50 minutes run of jobs using HPC scientific software Fluent and DL_POLY.

A. The System Under Test and Workload on SOL

The system under test (SUT) was a Sun Starfirex4170 1U rack mounted server with the following characteristics:

- CPU Processor speed, Type = 4-way 2.4GHz Sparc
- MEM Capacity = 8 GB
- NIO Network IO Capacity = 4x1Gb Ethernet
- kW Power rating = vendor predicts 83W to 149W for the 550W x 1400
- PSU = (940W or 760W in Vendor specification) 550W on nameplate

The initial Table of recorded process data can be seen in Figure 2, which is later converted into charts.

The system workload (WUT) DL-POLY [8] molecular dynamics and Fluent [9] Computational Fluid Dynamics applications can be seen in Figures 2 and 3.

Scan_ID	Location_or_Comments	RAM_Installed	RAM_Used_Kb	Total_Octets_IO	Power_U	PSU_Max
832	SOL node1 24/Sep DLPoly	8,063,056	979,780	697,052,978	336	550
831	SOL node1 24/Sep DLPoly	8,063,056	1,342,920	697,051,730	384	550
830	SOL node1 24/Sep DLPoly	8,063,056	1,342,920	697,051,730	384	550
829	SOL node1 24/Sep DLPoly	8,063,056	1,342,920	697,051,730	384	550
828	SOL node1 24/Sep DLPoly	8,063,056	1,342,920	697,051,730	384	550
827	SOL node1 24/Sep DLPoly	8,063,056	1,342,920	697,051,730	384	550
826	SOL node1 24/Sep DLPoly	8,063,056	1,342,920	697,051,730	384	550
824	SOL node1 24/Sep DLPoly	8,063,056	1,342,920	697,051,730	384	550
823	SOL node1 24/Sep DLPoly	8,063,056	1,342,880	697,051,730	384	550
822	SOL node1 24/Sep DLPoly	8,063,056	1,342,880	697,051,730	384	550
821	SOL node1 24/Sep DLPoly	8,063,056	1,342,780	697,051,730	384	550
820	SOL node1 24/Sep DLPoly	8,063,056	1,342,760	697,051,730	384	550
819	SOL node1 24/Sep DLPoly	8,063,056	1,342,760	697,051,730	384	550
818	SOL node1 24/Sep DLPoly	8,063,056	1,342,720	697,051,730	384	550
817	SOL node1 24/Sep DLPoly	8,063,056	1,342,520	697,051,730	384	550
816	SOL node1 24/Sep DLPoly	8,063,056	1,211,160	697,051,730	384	550
815	SOL node1 24/Sep fluent	8,063,056	979,780	696,913,262	312	550
814	SOL node1 24/Sep fluent	8,063,056	2,192,580	642,686,056	408	550
813	SOL node1 24/Sep fluent	8,063,056	979,780	561,891,156	336	550
812	SOL node1 24/Sep fluent	8,063,056	2,057,860	502,711,096	384	550
811	SOL node1 24/Sep fluent	8,063,056	979,760	422,025,374	312	550
810	SOL node1 24/Sep fluent	8,063,056	979,740	421,073,012	312	550
809	SOL node1 24/Sep fluent	8,063,056	979,740	421,073,012	312	550
808	SOL node1 24/Sep fluent	8,063,056	979,740	421,073,012	336	550
807	SOL node1 24/Sep fluent	8,063,056	979,740	420,115,818	336	550
806	SOL node1 24/Sep fluent	8,063,056	979,740	420,115,818	336	550
805	SOL node1 24/Sep fluent	8,063,056	2,112,640	389,523,300	408	550

Figure 1. Table of process data for DL-POLY and Fluent

Location: SOL node1 24/Sep fluent

appName	Values: Sum of NIOutil%	Sum of MEMutil%	Sum of CPUutil%
mpirun	2.32%	0.07%	0.01%
khugepaged	8.48%	0.00%	0.02%
fluent	6.97%	0.24%	0.03%
pbs_mom	8.48%	3.92%	0.08%
gmond	8.48%	5.60%	0.24%
snmpd	8.48%	0.77%	0.77%
cortex.13.0.0	2.32%	2.21%	4.16%
fluent.13.0.0	2.32%	1.31%	5.18%
glusterfs	16.96%	11.52%	8.24%
fluent_mpi.13.0	9.30%	4.48%	55.99%
Grand Total	**74.12%**	**30.12%**	**74.70%**

Figure 3. Workload Profile for Fluent

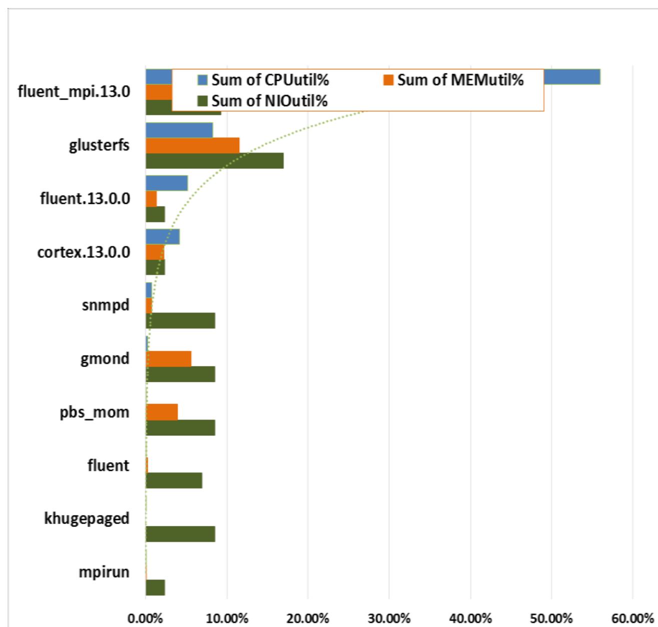


Figure 4. Workload Chart for Fluent

Out of 124 processes/applications the top 10 users of CPU are shown in the workflow chart in Figure 4.

The chart shows that processes use varying amounts of CPU and memory. Their combined activity makes up the Workload profile.

For capacity planning, the combined requirements of system resources from all these processes, is presented in the Survey chart, showing the overall demand during the survey. The data is generated during SNMP scans taken every 2 minutes and given a unique ID as shown on the horizontal axis. The 26 scans were collected over the 50 minutes.

Each workload resource can be shown as a percentage of the maximum value as in Figure 5.

It can be observed that the CPU utilisation climbs steadily, whilst memory utilisation peaks early and remains stable, and network I/O is almost negligible.

Figure 5 also illustrates the system utilisation and energy utilisation for two HPC applications. The first part of the experiment illustrates CPU, MEM, NIO and energy utilisation when Fluent is run on the system, whilst the second part of the survey shows the scans of the DL-POLY application. Even though the DL-POLY CPU utilisation level is higher than Fluent, the power drawn does not peak at a higher level. DL-POLY may be running more efficiently than Fluent in terms of MIPS per Watt.

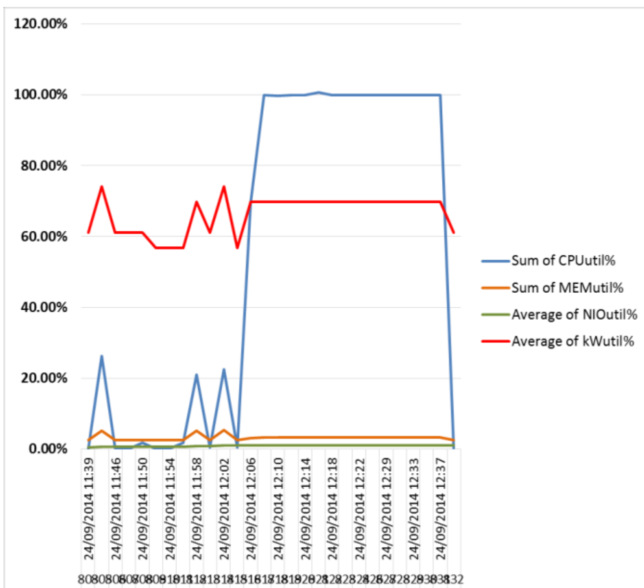


Figure 5.Survey Chart of Fluent and DL-POLY with Power Usage

V. IMTERACT TOOL EVALUATION IN GPU SYSTEM PROFILING

As energy usage becomes an increasing concern in data centres and high performance computing centres, it is becoming clear that software needs to exploit the available hardware to deliver results with a lower energy footprint.

The benefits of GPU processing have proven invaluable in this regard [10], a look at the top 500 list shows that the world's fastest supercomputers are using GPUs [11]. More and more locations for are investing in GPU accelerators; however the benefits of this hardware are not apparent without appropriately designed software. To this end, IMTeract tool was used to record the power draw of the machine whilst running different implementations of an FFT algorithm.

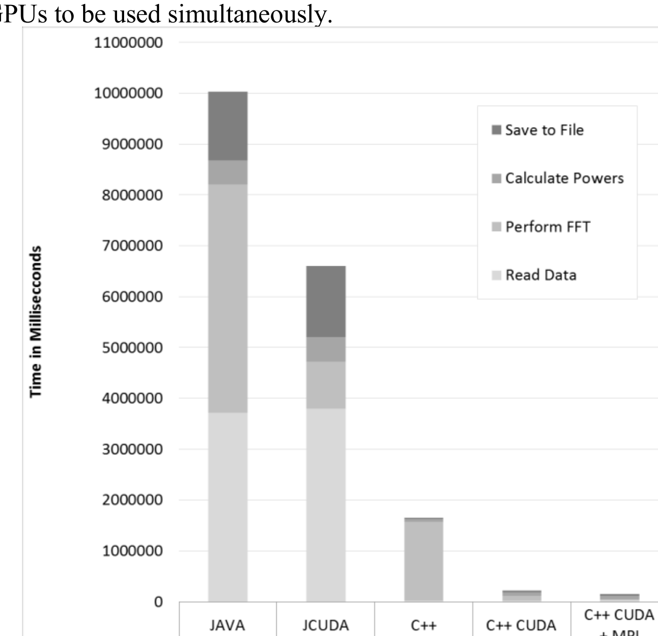
A. The system under test

The system under test is a GPU cluster consisting of a host machine containing 2 quad-core Intel Xeon processors running at 2.4 GHz, with 24GB RAM and a PCI-e expansion chassis containing 2 NVIDIA Tesla m2050 GPUs, each with 3GB RAM, and 448 stream cores running at 1.5 GHz. The current draw was measured using a rack monitoring device and two clamps on the power cabling to the host machine and to the GPU chassis.

B. The Workflow – FFT software

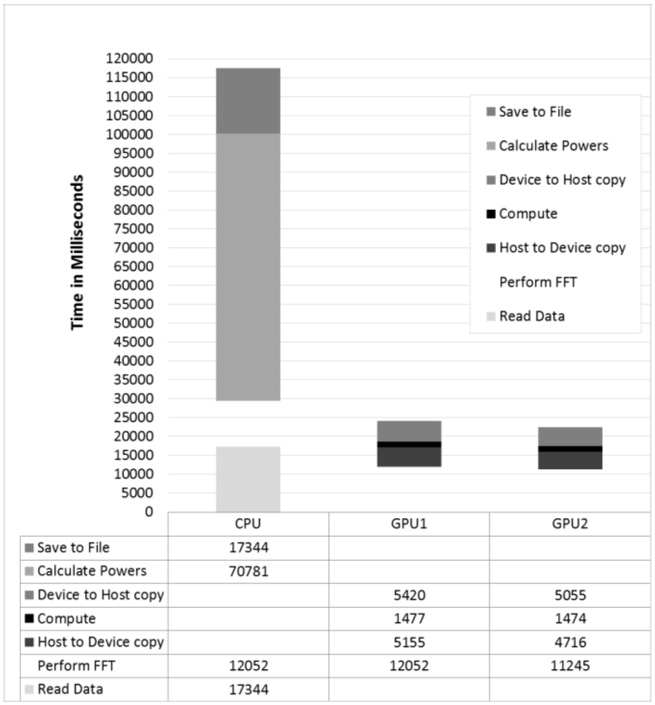
The system workload (WUT) is software for calculating a Fast Fourier Transform (FFT) of radio telescope data obtained from the SETI project[14]. A number of different versions of the software were tested using algorithms designed for a single machine (serial version) as well as versions designed for parallel execution on multiple GPUs. Each version of the FFT software reads data from a number of 2GB files, performs an appropriate number of FFTs, and saves the resulting data to a file. NVIDIA CUDA and CUFFT are used to provide GPU acceleration, and MPI is used to allow multiprocessing.

Each iteration of the software improves the performance in a number of ways. Initially, GPU acceleration was added using the JCUDA wrappers for JAVA [12]. In order to gain better access to CUDA functions, the software was rewritten in C++ which, together with using memory mapping for file access, was significantly faster than JAVA even without using the GPU. Finally, MPI was used to allow multiple GPUs to be used simultaneously.



	JAVA	JCUDA	C++	C++ CUDA	C++ CUDA + MPI
Save to File	1344062	1404548	29686	29524	34521
Calculate Powers	477218	483007	56436	70781	70781
Perform FFT	4491666	916464	1541246	90720	12052
Read Data	3713182	3801657	29686	29524	34521

Figure 6. Breakdown of processing time (ms)



	CPU	GPU1	GPU2
Save to File	17344		
Calculate Powers	70781		
Device to Host copy		5420	5055
Compute		1477	1474
Host to Device copy		5155	4716
Perform FFT	12052	12052	11245
Read Data	17344		

Figure 7. C++CUDA+MPI Method

Figure 6 shows running times for each iteration, broken into sections for each part of the program. Timing is measured by recording wall time within the program before and after each task, and additional GPU timing is provided by the NVIDIAVisual Profiler [13].

Figure 7 shows in more detail a breakdown of each parallel task in the fastest performing code which uses C++ CUDA code in parallel on two GPUs using MPI.

Hard disk read/write performance is reduced in the MPI version due to the fact that two processes are accessing the same data simultaneously, but this performance hit is far less significant that the performance gained by using two GPUs.

C. Energy Profiling

Each version of the software was run while the power usage was measured using the IMTeract tool. Peak draw for the system while running serial versions of the code was measured at 168 Watts. The power measured for parallel versions using GPUs was measured at 192 Watts. Using this data the power usage over the run time of the software can be calculated using (1)

$$E(kWh)=Power(W)*Time(hours)/1000. \quad (1)$$

Figure 8 shows estimated energy usage of software performing 244 FFF calculations.

By making use of GPU hardware and software methods to significantly reduce processing time, it has been possible to perform the same task using a fraction of the energy. Even though power usage increases as GPUs are incorporated, as they can allow tasks to be completed much quicker, much less power is used over the course of the task.

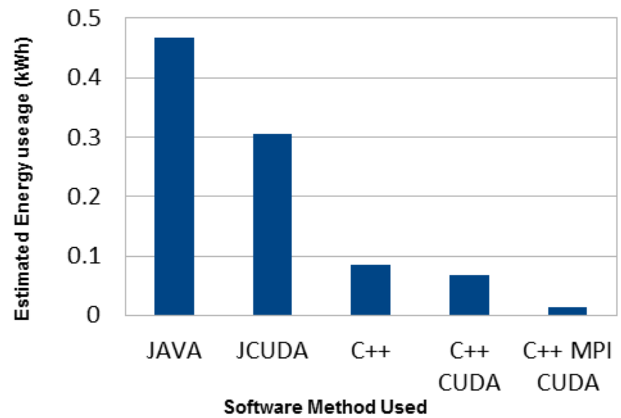


Figure 8. Code Power Efficiency for software performing 244 FFTs

There is still scope for further efficiency improvements however. The GPUs are only utilised for a small amount of the total running time of the software which can be observed using the NVIDIA Visual Profiling tool, with activity in very brief periods interspaced with large periods of idle time as show in Figure 9.

GPU utilisation could be improved by parallelising the remaining serial calculation still carried out on the CPU, the spectrum calculation. This would further reduce total running time as well as improving GPU utilisation. Additionally, in the final version of the software (C++CUDA+MPI), disk access comprises the remainder of the running time. Upgrading to solid state storage would not only improve instantaneous power draw, but would also improve the efficiency of software, allowing the GPU to work on data with greater frequency rather than idling while waiting for data.

This set of experiments demonstrated that the IMTeract tool can be used alongside existing software tools to assist in energy efficiency profiling of different system architectures and applications.

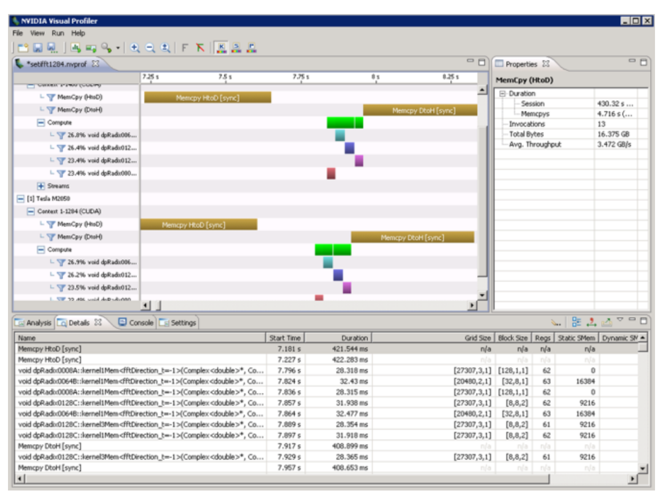


Figure 9. NVIDIA Profiling Tool

VI. CONCLUSION

The aim of our project was to build a heterogeneous and non-invasive tool which would show energy cost by server, storage and networking infrastructure, and application data. We have demonstrated that it is possible to build a tool to look at equipment sited within a typical data centre and to measure their energy efficiency.

The tool is an improvement on the existing proprietary tools because it is non-proprietary, in situ, and non-invasive and will enable measurement of infrastructure energy costs.

The IMTeract tool is capable of measuring energy cost of all data centre hardware, not just the servers. It will help data centre managers to make predictions about what would happen with certain configurations of infrastructure equipment and applications.

The IMTeract tool can be used to benchmark applications and processes and to understand the interplay of CPU/Memory Disk I/O and Networking I/O and to extrapolate when one of those resources would impinge on the best working of the application.

The tool can collect SNMP data at intervals to help understand the system and application in terms of CPU, Disk & Network I/O, Memory, and Power.

Power information is collected about an infrastructure component (server, storage or network) and can be used for modelling application requirements for power, CPU, Memory, Disk and Network I/O by apportioning the power used across the running processes. Power usage and network traffic can be allocated to the processes seen, and the tool infers the energy use as being proportional to the CPU usage.

We have completed non-invasive tests using the prototype tool in commercial and academic data centres using different applications. The tool was used to profile and measure power of components in the SOL Sun cluster and assess energy efficiency of Fluent, DL-POLY software; and the VEGA GPU cluster running different FFT algorithms.

Whilst we recognised that we would be looking at established data centres and the equipment held there, we appreciate that the more recent servers/storage/networking products from the vendors now contain the information to allow decision making on the energy efficiency of the equipment. However, we could now address all of the older equipment across the Data Centre estate. This old infrastructure does not have embedded monitoring for its energy use, and so our tool gives access to the data, allowing data centres to be utilised better.

The project has demonstrated a way to benchmark running applications, model what their extremes are, model their exhaustions and finally, to allow data centre resources to be fully utilised. This will save energy.

ACKNOWLEDGMENT

Our thanks to Innovate UK for funding the project and The University of Huddersfield for the access to the Data Centre 2 and QGG Campus Grid.

REFERENCES

[1] Hamacher, A., Lewis,B. KTN Magazine. Energy Efficient Computing, 2014.

[2] Digital Europe, www.digitaleurope.org (accessed April 2015)

[3] Innovate UK (Technology Strategy Board),. 2013. https://www.gov.uk/government/organisations/innovate-uk

[4] Raritan, DCIM Software, http://www.raritan.com/ap/products/dcim-software (accessed April 2015)

[5] Munnings, D., Green Computing and Data Centre Efficiency, Master by Research thesis, The University of Huddersfield (publication pending)

[6] Eagle-I Rack Monitoring Unit, President, http://www.apwpresident.com/products/datacenter-environment-monitoring/eagle-i-%E2%80%93-rack-monitoring-unit (accessed April 2015).

[7] Holmes, V. and Kureshi, I. (2010) 'Huddersfield University Campus Grid: QGG of OSCAR Clusters' Journal of Physics: conference series , 256 (1), p. 012022. ISSN 1742-6596

[8] I.T. Todorov, W. Smith, K. Trachenko & M.T. Dove, DL-POLY_CLASSIC, Journal of Materials Chemistry, (2006) 16, 1911-1918, http://www.ccp5.ac.uk/DL_POLY_CLASSIC/ (accessed April 2015).

[9] ANSYS, FlUENT, http://www.ansys.com/Products/Simulation+Technology/Fluid+Dynamics/Fluid+Dynamics+Products/ANSYS+Fluent (accessed April 2015).

[10] On the Performance and Energy-efficiency of Multi-core SIMD CPUs and CUDA-enabled GPUs. R. Duarte et. Al. IEEE International Symposium on Workload Characterization, 2013.

[11] top500. www.top500.org [accessed April 2015]

[12] Newall, M., Holmes, V., Venters, C. and Lunn, P. (2015) 'Delivering Faster Results Through Parallelisation and GPU Acceleration'. In: Intelligent Systems in Science and Information 2014. Switzerland: Springer International Publishing . pp. 309-320. ISBN 978-3-319-14654-6

[13] NVIDIA Visual Profiler, https://developer.nvidia.com/nvidia-visual-profiler (accessed April 2015)

[14] [14] SETI@home, http://setiathome.ssl.berkeley.edu/ (accessed April 2015)

[15] http://www.gov.uk

[16] http:www.gartner.com

[17] http://www.ganglia.sourceforge.net

[18] https://www.nagios.com

[19] http://www.intel.com

[20] Rong Gee, Xizhou Fang, Shuaiwen (et. al), 'PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications'. In: IEEE Transactions on Parallel and Distributed Systems, Vol 21, NO. 5. 2010.

[21] http://www.mageec.org

[22] Thial, J.(2006), 'An overview of software performance analyzis tools techniques', Washington University in St. Louis, Tech. Rep. 2006

[23] Intel Corporation, 'Intel® VTune™Performance Analyzer 9.1 for Linux*', https://software.intel.com/sites/default/files/m/c/9/e/3/7/6741-vtune_release_notes_linux.pdf

[24] Luk, C., et. al., "Pin: Building Customized Program Analysis Tools with Dynamic Instrumentation," Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI '05).

[25] Microsoft Research, 'Joulemeter: Computational Energy Measurement and Optimization', http://research.microsoft.com/en-us/projects/joulemeter/default.aspx