Jean-Marc Pierson
Georges Da Costa
Lars Dittmann (Eds.)

LNCS 8046

Energy Efficiency in Large Scale Distributed Systems

COST IC0804 European Conference, EE-LSDS 2013
Vienna, Austria, April 2013
Revised Selected Papers

www.cost804.org
Action IC0804

Springer

Jean-Marc Pierson · Georges Da Costa
Lars Dittmann (Eds.)

Energy Efficiency in Large Scale Distributed Systems

COST IC0804 European Conference, EE-LSDS 2013
Vienna, Austria, April 22-24, 2013
Revised Selected Papers

Springer

Editors
Jean-Marc Pierson
Georges Da Costa
University of Toulouse
Toulouse
France

Lars Dittmann
Technical University of Denmark
Lyngby
Denmark

ISSN 0302-9743 ISSN 1611-3349 (electronic)
ISBN 978-3-642-40516-7 ISBN 978-3-642-40517-4 (eBook)
DOI 10.1007/978-3-642-40517-4
Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2013948856

CR Subject Classification (1998): H.2, C.2, J.1, C.4, D.4, K.6

LNCS Sublibrary: SL 5 – Computer Communication Networks and Telecommunications

Preface

The EE-LSDS ("Energy Efficiency in Large-Scale Distributed Systems") conference attracted high-quality contributions in the area of energy-efficient communication, cloud computing, and high-performance computing. The particular originality of the conference is to address large-scale systems. It served as the final event of the COST Action IC0804 (www.cost804.org).

The purpose of the conference was to present some of the results obtained during the course of the COST Action IC0804 as well as external papers. Dedicated to energy efficiency in large-scale distributed system, the COST Action lasted 4 years, from May 2009 to May 2013. With about 150 researchers involved, and more than 40 institutions coming from 27 countries in Europe and beyond, it has been a tremendous open instrument to motivate and build bridges between communities. Hopefully researchers from around the world will find inspirations in all the Joint publications from the Action (more than 75 publications) to date; some of them are in the volume you have in your hand (six are Joint works).

Organized at the University of Vienna, during April 22–24, 2013, the call for papers attracted 31 contributions for all sessions (17 for the full paper session). A thorough review process was conducted by the 41 members of the Technical Program Committee, as well as additional external reviewers, with two to four independent reviews per submission. The Program Committee selected 15 full paper presentations. Additionally, a Work in Progress and a Demo Session were organized, attracting, respectively, seven short papers and three demonstrations for this volume. The program of the conference was augmented by a special session showing results from three selected Short-Term Scientific Missions (collaborative works between partners of the Action) conducted within the Action, and by a society session where the Eco4Cloud company, a spinoff originating from ideas of the Action was presented. Altogether, about 70 participants were registered at the conference.

In this volume, the papers are organized among different sections, reflecting the diversity of the topics addressed and the richness of the discussions held at the conference: Modeling and monitoring of power consumption (six contributions), distributed, mobile, and cloud computing (six contributions), HPC computing (four contributions), wired and wireless networking (seven contributions), and standardization issues (two contributions).

The final result would not have been possible without the invaluable contribution of a number of colleagues. We would like to acknowledge the great work of the local organizer Helmut Hlavacs, from the University of Vienna. We also express our gratitude to members of the Technical Program Committee and all external reviewers, for their competence and dedication. Finally, our most sincere thanks to all authors, who ultimately made the conference a successful and stimulating event.

Jean-Marc Pierson
Georges Da Costa
Lars Dittmann

Organization

General Chair: Jean-Marc Pierson
Program Chair: Georges Da Costa and Lars Dittmann
WiP Chair: Chris Phillips
Demo Chair: Anne-Cécile Orgerie
Local Organizer: Helmut Hlavacs

Technical Program Committee

Andrea Passarella
Anhtuan Trinh
Anne-Cécile Orgerie
Ariel Oleksiak
Carlos Juiz
Chris Phillips
Domenico Talia
Erez Kantor
Eugen Feller
Georges Da Costa
Giuseppe Anastasi
Hans-Arno Jacobsen
Helmut Hlavacs
Henri Casanova
Hermann Demeer
Ivona Brandic
Jean-Marc Menaud
Jean-Marc Pierson
Johan Lilius
John Morrison
Jukka Manner
Karinanna Hummel
Lars Dittmann
Laurent Lefevre
Louis-François Pau
Luis Neves
Markus Fiedler
Mirek Klinkowski
Ove Morck
Oznur Ozkasap
Paolo Monti
Paolo Trunfio
Phuoc Tran-Gia
Rizos Sakellariou
Schahram Dustdar
Sebastien Varrette
Shay Kutten
Shrisha Rao
Torsten Braun
Vlastimir Glamocanin
Zhiyi Huang

Contents

Part I

Modeling and Monitoring of Power Consumption

Solving Some Mysteries in Power Monitoring of Servers: Take Care of Your Wattmeters!

Mohammed El Mehdi Diouri[1](✉), Manuel F. Dolz[2], Olivier Glück[1], Laurent Lefèvre[1], Pedro Alonso[3], Sandra Catalán[2], Rafael Mayo[2], and Enrique S. Quintana-Ortí[2]

[1] INRIA Avalon Team, Laboratoire de l'Informatique du Parallélisme, UMR CNRS 5668, ENS Lyon, INRIA, Université Lyon 1, Lyon, France
{mehdi.diouri, olivier.gluck, laurent.lefevre}@ens-lyon.fr

[2] Depto. de Ingeniería y Ciencia de Computadores, Universitat Jaume I, 12071 Castellón, Spain
{dolzm, catalans, mayo, quintana}@uji.es

[3] Depto. de Sistemas Informáticos y Computación, Universitat Politècnica de València, 46022 Valencia, Spain
palonso@dsic.upv.es

Abstract. Large-scale distributed systems (e.g., datacenters, HPC systems, clouds, large-scale networks, etc.) consume and will consume enormous amounts of energy. Therefore, accurately monitoring the power and energy consumption of these systems is increasingly more unavoidable. The main novelty of this contribution is the analysis and evaluation of different external and internal power monitoring devices tested using two different computing systems, a server and a desktop machine. Furthermore, we also provide experimental results for a variety of benchmarks which exercise intensively the main components (CPU, Memory, HDDs, and NICs) of the target platforms to validate the accuracy of the equipment in terms of power dispersion and energy consumption. This paper highlights that external wattmeters do not offer the same measures as internal wattmeters. Thanks to the high sampling rate and to the different measured lines, the internal wattmeters allow an improved visualization of some power fluctuations. However, a high sampling rate is not always necessary to understand the evolution of the power consumption during the execution of a benchmark.

Keywords: Wattmeters · Energy and power analysis · Power profiling · Servers

1 Introduction

For decades, the computer science research community exclusively focused on performance, which resulted in highly powerful, but in turn, low efficient systems with a very high total cost of ownership (TCO) [1]. Yet, in recent years, the HPC community has acknowledged that the energy efficiency of HPC systems is a major concern in designing future exascale systems [2,3].

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 3–18, 2013.
DOI: 10.1007/978-3-642-40517-4_1,

Nowadays there exists a strong research effort towards energy-efficient supercomputers. Hardware provides part of the solution by exposing unceasingly more energy-efficient devices which also provide abilities that current operating systems can successfully leverage to save energy [4]. Mechanisms such as Dynamic Voltage Scaling (DVFS) or P-state management have also been used to develop power-aware user-level software [4–6].

The Green500 list seeks to raise the awareness of power and energy consumption in supercomputing by reporting the power dissipation and energy efficiency of large-scale HPC facilities. Even the Top500 list is currently tracking the power draw by today's most powerful HPC systems ranking their efficiency in terms of Mflops per Watt [7]. The metric used to build the Green500 List is limited by the use of LINPACK benchmark for performance/energy measurement, because this test primarily stresses the CPU component of an HPC system [7]. Clearly, a more elaborate figure is needed to inspect and understand all the power sinks in a computing node. Some proposals obtain the consumption of different parts of the node (CPU, memory, disks, fans, ...) but they are circumscribed to either one [8] or a few benchmarks types [9]. Furthermore, the most prevailing infrastructure comprises only an external power meter [1,10]. Another issue arises on how to process the power/energy samples due to the high variability to which they are subject to. Only some contributions deepen to obtain, e.g., a statistical regression on the samples to produce reliable results [11]. Given the foregoing, we contribute in this paper the following:

- We target five wattmeters (external and internal) to analyze two different systems representative of current general-purpose platforms: a desktop computer and a server node.
- In order to evaluate the precision of the data acquisition devices, we use and deploy five different types of benchmarks which stress different components of the system.
- We use a framework of easy-to-use and scalable tools to analyze the power variability and the energy consumption which comprises, among others, the `pmlib` library to interact with power measurement units [12].
- Results are displayed using *boxplots*. This graphically depicts five-number groups of data that illustrate the variability of the samples which may be highly affected by environmental conditions such as temperature fluctuation.

The rest of the paper is structured as follows. In the next section, we describe the experiment setup. The energy consumption obtained with the wattmeters using all the benchmarks is analyzed in Sect. 3. In Sect. 4, we discuss in more detail these results by processing samples under a statistical model based on boxplots. Section 5 performs an additional analysis by varying the wattmeters sampling rate. The paper is closed with a section that contains a discussion and a few conclusions.

2 Experimental Setup

This section describes the power measurement devices, the power measurement framework, the target platforms, and the benchmarks used in our evaluation.

Power Measurement Devices. We classify the measurement devices into two main types: external AC meters, which are directly attached to the wires that connect the electric socket to the computer Power Supply Unit (PSU); and the internal DC meters, responsible for measuring the output wires leaving the PSU that energize the components of the mainboard. Table 1 presents in detail the specifications of the wattmeters that we used.

Table 1. Specifications of the wattmeters.

	External AC		Internal DC		
Wattmeter	OmegaWatt	WattsUp	PowerMon2	NI	DCM
Manufactured by	OmegaWatt[a]	WattsUp?[b]	RENCI iLab[c]	National Instruments[d]	Universitat Jaume I
# Channels	6	1	8	32	12
Channel type	Standard power PC cord	Standard power PC cord	All ATX-related lines (3.3 V, 5 V and 12 V)[e]	12 V ATX-related lines	12 V ATX-related lines
Power nature	Average	Average	Instantaneous	Instantaneous	Instantaneous
Microcontroller	-	-	Atmel ATmega16	NI9205 NIcDAQ-9178	*Microchip* PIC 18
Power sensors	-	-	*Analog Devices ADM1191* resistors	*LEM HXS 20-NP* transducers	*LEM HXS 20-NP* transducers
Sampling Rate (S/s) per channel	1	1	1024[f]	1000	28
Accuracy	$< \pm 1\%$	$< \pm 1.5\%$	$\pm 5\%$	$\pm 1\%$	$\pm 1\%$
Interface	RS232	USB	USB	USB	RS232
Price	600 €	200 €	125 €	2700 €	Not commercialized

[a] OmegaWatt: http://www.omegawatt.fr/
[b] WattsUp: https://www.wattsupmeters.com/
[c] PowerMon2: http://ilab.renci.org/powermon
[d] NI: http://www.ni.com/
[e] 3.3 V and 5 V lines measure the power consumption of some components of the mainboard (GPUs, NICs, etc.) while 12 V lines measure the power consumption of the CPUs and fans.
[f] For PowerMon2, we used only 100 S/s.

Power Measurement Framework. The `pmlib` software package is developed and maintained by the HPC&A research group of the Universitat Jaume I to investigate power usage of HPC applications. The current implementation of this package provides an interface to utilize all the above-mentioned wattmeters and a number of tracing tools. Power measurement is controlled by the application using a collection of routines that allow the user to query information on the power measurement units, create counters associated to a device where power data is stored, start/continue/terminate power sampling, etc. All this information is managed by the `pmlib` server, which is in charge of acquiring data from the devices and sending back the appropriate answers to the invoking client application via the appropriate `pmlib` routines (see Fig. 1).

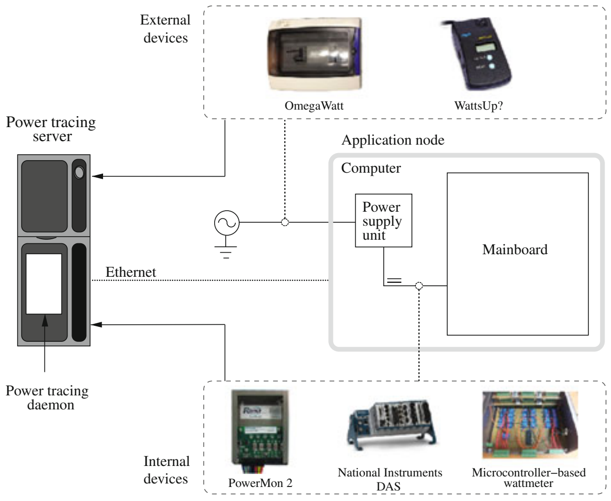


Fig. 1. Single-node application system and sampling points for external and internal wattmeters.

Target Platforms. The analysis and evaluation made with the wattmeters has been carried out on two different platforms: a desktop platform and a server node. The desktop computer consists of an Intel Ivy Bridge Core i7-3770K equipped with 4 cores running at 3.50 GHz and 16 GB of RAM. We will denote this machine as INTEL_DESKTOP. The server machine, referred to as AMD_SERVER, integrates 4 AMD Opteron 6172 of 12 cores (total of 48 cores) running at 2.10 GHz and contains 256 GB of RAM.
Benchmarks. To evaluate the energy and power behavior, we run different types of workloads to provoke and encourage the use of specific parts of the platforms. CPU, memory, NICs and HDDs are the main components we stress in our experiments. To achieve this purpose, we selected the following specific benchmarks:

– `idle`: This benchmark employs the `sleep` POSIX routine[1] to suspend processor activity, thus generating idle periods that let the hardware promote the cores to low power consumption states, also known as C-states[2].
– `iperf`[3]: This tool performs network throughput measurements. It can test either TCP or UDP throughput. To perform an `iperf` test, we set both a server and a client. Since the package features a large number of options, we only measure this tool running it as TCP client.
– `hdparm`[4]: This application provides a command line interface to various kernel interfaces supported by the Linux SATA/PATA/SAS *libATA* subsys-

[1] `sleep`: http://linux.die.net/man/3/sleep
[2] Advanced Configuration and Power Interface. Revision 5.0. http://www.acpi.info/
[3] `iperf`: http://iperf.fr
[4] `hdparm`: http://linux.die.net/man/8/hdparm

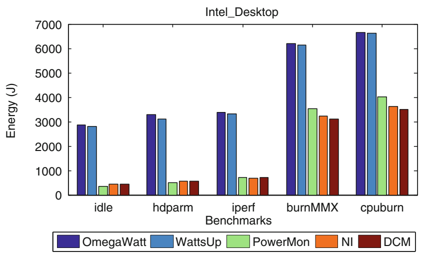
tem. We use the `-t` option to perform timings on device reads and to stress the HDD.

- `cpuburn`[5]: This benchmark heats up any CPU to the maximum operating temperature that is achievable using ordinary software. We map `cpuburn` processes into specific cores in order to measure the power when different number of cores are used.
- `burnMMX`[6]: This program, included in the `cpuburn` package, specifically stresses the cache and memory interfaces. `burnMMX` processes are mapped into specific cores in order to measure the power when different number of cores are used.

3 Energy Consumption Analysis

In this section we analyze the variability and accuracy of the external and internal wattmeters using all benchmarks introduced in the previous section on the two selected machines: Intel_Desktop and AMD_Server.

Figure 2 presents the energy consumption measured from the execution of the benchmarks, during 60 seconds each, on both platforms. Bars labeled as `idle` represent the energy consumption when leaving the platform doing nothing for a full minute. Bars `hdparm` and `iperf` report the energy registered by wattmeters when, respectively, the HDDs and the NICs are stressed. For the `burnMMX` and `cpuburn` benchmarks we heat all the cores by mapping one process per core. Specifically, we use the `taskset` command to bind processes to the cores of the machines. It is important to note that, before taking measures with these two benchmarks, we warm up the machines by running the corresponding tests up to the maximum temperature ($\approx$ 10 min.). We represent the aggregated energy consumption calculated as the addition of energy measurements in all the 12 V lines. Even though PowerMon2 can also measure the 3.3 V and 5 V lines, we rather prefer to account for the 12 V lines only, to provide a fair comparison with other internal wattmeters that are only able to measure 12 V lines.



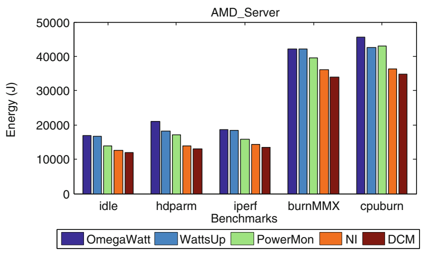


Fig. 2. Extra energy consumption of the benchmarks measured with the wattmeters.

[5] `cpuburn`: http://manpages.ubuntu.com/manpages/precise/man1/cpuburn.1.html
[6] `burnMMX`: http://pl.digipedia.org/man/doc/view/burnMMX.1

Figure 2 shows that the energy consumptions registered with both external wattmeters (OmegaWatt and WattsUp) are very similar. However, we observe a different scenario for the internal wattmeters. Indeed, in the light of the energy measured by these devices, it is easy to observe that the values provided by PowerMon2 are almost always higher than those registered by NI and DCM. These variations are mainly due to the use of different components to measure voltage/current of the internal wires. The differences between internal wattmeters, sometimes significant, are also due to the large amount of samples per second taken from the lines. DCM works at 28 samples per second (S/s), but PowerMon2 and NI respectively sample at 100 S/s and 1,000 S/s, and rapid variations are not captured by low sampling devices.

In contrast to the external wattmeters, the internal devices measure power consumption downstream the PSU. Thus, the internal measurements do not account for the PSU and other components like HDDs and/or GPUs. This explains why the energy consumption registered by the internal wattmeters is lower than that registered by the external wattmeters as it is easily observed in the graph for all the benchmarks regardless of the target machine. Providing an internal measurement in addition to an external measurement can inform us about the inefficiency of the PSU which may be different from a node to another. Compared to Intel_Desktop, this difference between external and internal wattmeters seems to be relatively less significant for AMD_Server. This is mainly due to the fact that AMD_Server is composed of 48 cores and several fans whose power consumption is included in the internal power measurements and represents a significant part of the total consumption of the machine.

We also studied if the energy fluctuations observed in Fig. 2 were related to the wattmeter calibration. Figure 3 shows the extra energy consumption corresponding to the same experiments reported in Fig. 2, but removing the energy consumption of the `idle` benchmark from the energy consumption of each benchmark. While we could expect small or no differences between the different measurement devices, some variations appear. Specifically, in some cases there are still differences that range from 2.23% to 65.75%.

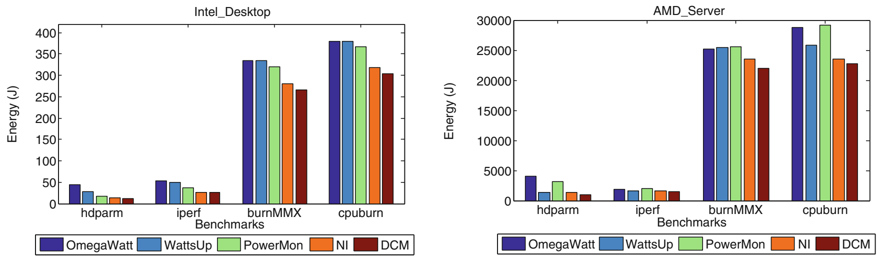


Fig. 3. Extra energy consumption of the benchmarks (i.e. without the `idle` part) measured with the wattmeters.

4 Power Consumption Analysis

In this section, we analyze the power measurements made by the different considered wattmeters. Through this analysis, we first seek to show the variability of the power measurements for different workloads running on the two different target platforms, Intel_Desktop and AMD_Server. By comparing distinct wattmeters and different machines, we intend to identify the impact of measuring the power consumption internally or externally with a variable measurement frequency.

At this point, we consider the following scenarios: the target machine is `idle` (Fig. 4), one core of the machine runs `hdparm` (Fig. 5), and all cores of the machine run `cpuburn` (Fig. 6). We execute each benchmark on the two different target machines, and measure the power consumption during 60 seconds using the external wattmeters OmegaWatt and WattsUp, and the internal wattmeters PowerMon2, NI and DCM.

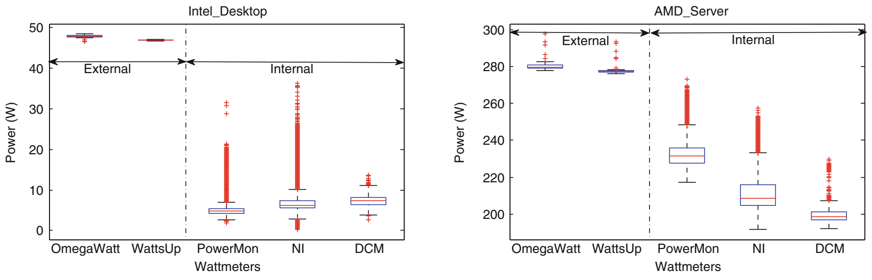


Fig. 4. Dispersion of the power consumption measurements for benchmark `idle`.

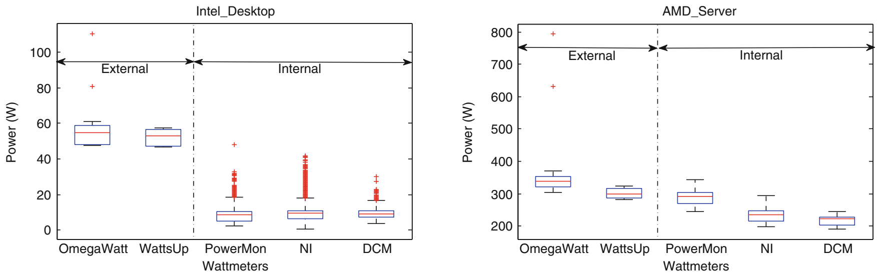


Fig. 5. Dispersion of the power consumption measurements for benchmark `hdparm`.

Figures 4, 5 and 6 present boxplots showing the distribution of the power consumption measurements from this experiment. Each boxplot graphically depicts groups of numerical data using a five-number summary: the smallest observation

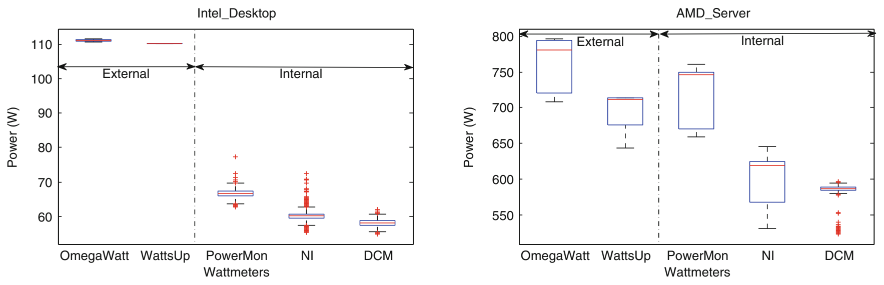


Fig. 6. Dispersion of the power consumption measurements for benchmark `cpuburn`.

(sample minimum), lower quartile (Q1), median (Q2), upper quartile (Q3), and largest observation (sample maximum). The plots also indicate which measurements, if any, should be considered outliers. Since internal measurements do not take into account the inefficiencies of the PSU and even some components, the corresponding boxplots are, in most cases, below those corresponding to external wattmeters.

Alike in the previous experiment, with energy measurements, the boxplots for INTEL_DESKTOP show that the variability of the power measurements obtained from external devices are similar, independently of the benchmark considered. This also holds for the internal power measurements, except for `cpuburn` where POWERMON2 provides slightly higher power measurements compared with the other wattmeters. As concerns AMD_SERVER, the differences are more visible: neither the external nor the internal power measurements are concordant, particularly for the `cpuburn` benchmark. Indeed, for all the benchmarks running on AMD_SERVER, power measurements from POWERMON2 are always higher than those provided by the other internal wattmeters. Furthermore, compared with WATTSUP, OMEGAWATT provides higher power measurements, especially for the `cpuburn` and `hdparm` benchmarks. These differences may be due to the fact that these wattmeters are made of different components offering different degrees of accuracy, especially when applied to measure too high power consumptions (more than 300 W).

What is even more mysterious is that when benchmark `cpuburn` is run on AMD_SERVER, POWERMON2 registers higher power measurements compared to WATTSUP, even when POWERMON2 power measurements do not take into account the PSU inefficiencies and some internal components. This also holds for some power samples when the `hdparm` runs on AMD_SERVER. We suspect that POWERMON2 is less accurate when used to measure too high power consumptions (more than 500 W).

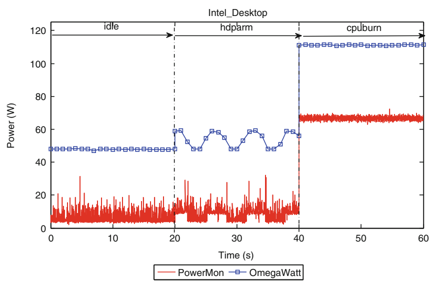
Furthermore, in Fig. 4 we notice that, compared to the external measurements, the results captured with the internal devices are more dispersed and generate many outliers. This is certainly due to the high sampling frequency of internal wattmeters that allow to measure some power fluctuations that the external wattmeters cannot capture. We notice from Fig. 5 that, contrary to WattsUp, OmegaWatt registers some strange outliers while measuring the `hdparm` benchmark. This can be due to the high variability of the power consumption during the execution of the `hdparm` benchmark provoking that OmegaWatt is occasionally not accurate. Moreover, Fig. 6 shows that when `cpuburn` runs on AMD_Server the power measurements are highly dispersed: a variability of almost 100 W for OmegaWatt, PowerMon2, NI; and about 70 W for WattsUp and for DCM. This high power variability for the `cpuburn` on AMD_Server suggests that this benchmark is fluctuating too much. By analyzing the power profiles in Sect. 5, we expect to confirm the fluctuation in `hdparm` and `cpuburn` when they are executed on AMD_Server.

5 Power Profile Analysis

In this section we continue the behavior analysis of the set of benchmarks on the target machines. In particular, we show power profiles, depict the impact of reducing the sample rate of wattmeters and, finally, analyze the power transferred by each power line using PowerMon2. Analyzing the power profiles of an application is a step beyond studying its energy and power consumption.

5.1 Analysis of the Power Profiles

Let us start by considering Figs. 7 and 8 which, respectively, represent the profiles on Intel_Desktop and AMD_Server of 20 seconds of the execution of benchmarks `idle`, `hdparm`, and `cpuburn` obtained from both external and internal wattmeters. The plot in the left-hand side of Fig. 7 depicts the power trace when OmegaWatt and PowerMon2 (sampling only the 12 V lines) are simultaneously connected to Intel_Desktop; the plot in the right shows the same information but replacing the wattmeters by WattsUp and NI. The aim of this comparison is to inspect the same scenario with two different power measurement devices configurations. The results show that the power profile from the external wattmeters are nearly the same; however, for the internal devices some variations exist. The first slight drop of power that PowerMon2 provides when compared



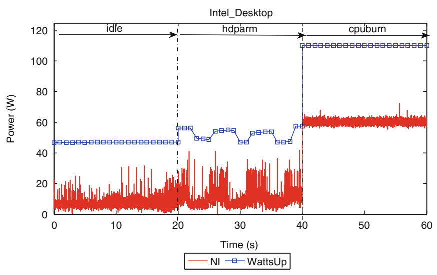


Fig. 7. Power profiles obtained when running benchmarks `idle`, `hdparm` and `cpuburn` benchmarks on Intel_Desktop.

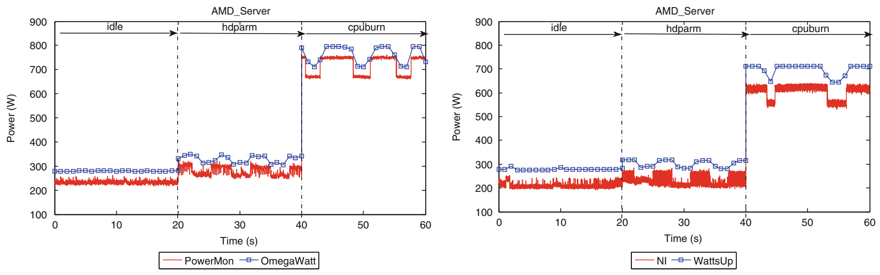


Fig. 8. Power profiles obtained when running benchmarks `idle`, `hdparm` and `cpuburn` benchmarks on AMD_Server.

with the NI; this observation was already made in the experiments of Sects. 3 and 4. Apart from that, we observe much more noise with NI; nevertheless this behavior is due to the high sampling rate of this device. These comparisons demonstrate that it is easy to observe how these two scenarios using different wattmeters provide reliable power profiles. Our measurements could be displaced along time due to the high frequency. However, by plotting power profiles from internal wattmeters, we are interested in analyzing the internal behavior of the applications: for example, to detect some special power increases that could be filtered by external wattmeters.

Drawing attention to Fig. 8, we observe how the external wattmeters OmegaWatt and WattsUp provide different power profiles for `hdparm` and, more acutely, `cpuburn`. These differences mainly come from the specifics of the devices and from potential environment changes (e.g., room temperature), as these experiments could not be performed simultaneously. (We were not able to connect two external or internal wattmeters at the same time.) The same situation occurs for the internal wattmeters with the PowerMon2 profile being highly displaced from that obtained from NI.

We also highlight the spikes and drops observed in the power profile when running the `cpuburn` benchmark on AMD_Server. Remember that this platform contains 48 cores which we fully populated with this kind of processes. We relate this behavior to the BIOS-mainboard settings and fans, that are constantly on and off in order to cool the machine and maintain the platform's temperature at a constant level.

5.2 Impact of the sample rate: the NI case

In this section, we investigate whether a very high sample rate (more than 100 S/s) produces power fluctuations that are not observed with a low sample rate (1 S/s). For this purpose, we measure the power consumption with NI and a frequency of 1,000 S/s during 30 seconds for `hdparm` and `cpuburn` on Intel_Desktop and AMD_Server. Then, we reduce the sample rate by

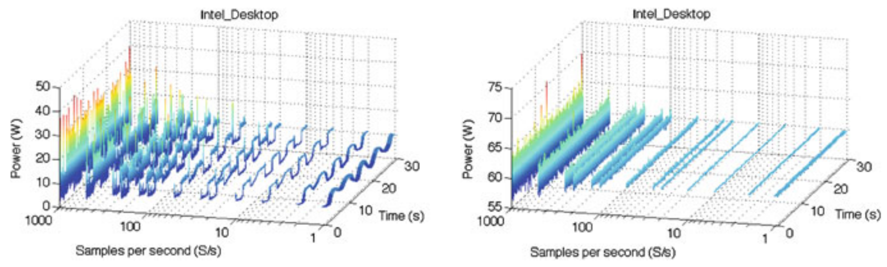


Fig. 9. Power profiles obtained with NI for `hdparm` (left side) and `cpuburn` (right side) running on INTEL_DESKTOP with a configurable sample rate.

applying the formula:

$$S_j^r = r\frac{\sum_{i=1+1000(j-1)/r}^{\frac{1000j}{r}}(S_i^{1000})}{R}, \tag{1}$$

where r is the reduced sample rate, R is the original sample rate, S_i^{1000} is the i^{th} sample taken with 1,000 S/s and is S_j^r the j^{th} sample taken with r as sample rate.

Figure 9 shows the power profiles obtained with NI by reducing the sample rate from 1,000 S/s to 1 S/s for `hdparm` (left side) and `cpuburn` (right side) respectively on INTEL_DESKTOP. From this figure, we notice that the noise induced by the high sample rate (more than 200 S/s) masks the spikes and drops of `hdparm` making them harder to perceive. However, a reduced sample rate (less than 50 S/s) hides some interesting power fluctuations, like the high spikes that we can observe just before each drop when the sample rate is 50 S/s. With respect to the `cpuburn` benchmark, we can observe that the power fluctuates between 57 W and 63 W when the sample rate is 1,000 S/s. The shape of the power profile becomes thinner when reducing the sample rate. Below 50 S/s, we notice that the power profile is a constant line devoid of noise. Thus, for INTEL_DESKTOP, we may need a medium sample rate (between 50 S/s and 200 S/s) for a benchmark like `hdparm`, while a low sample rate (1 S/s) is enough to observe the power profile for a benchmark like `cpuburn`.

The same experiment was repeated in AMD_SERVER and results are given in Fig. 10. In this case, we notice a behavior for `hdparm` similar to that observed on INTEL_DESKTOP. On the other hand, the behavior is different for AMD_SERVER for the `cpuburn` benchmark. Indeed, contrary to what we observed for `cpuburn` running on INTEL_DESKTOP, a medium sample rate (between 50 S/s and 200 S/s) helps to understand better the spikes and drops that appear when `cpuburn` is executed this platform. With 1 S/s, we still perceive the spikes and drops, which confirms that they correspond to real power fluctuations on AMD_SERVER which are not simply due to the noise that may be generated by a high sampling rate.

In summary, measuring at a very high sample rate (500 S/s) is not always necessary for power profiling applications and even may cause some noise that

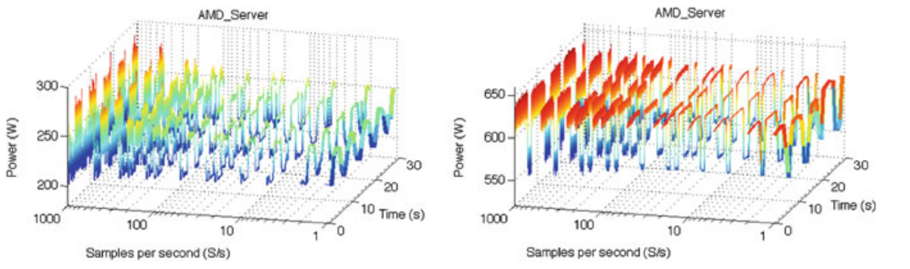


Fig. 10. Power profiles obtained with NI for `hdparm` (left side) and `cpuburn` (right side) running on AMD_Server with a configurable sample rate.

masks the general shape of the power profile. The best sampling rate is not always the highest one, but the one that best enables to understand fluctuations.

5.3 Internal Channel Analysis: The PowerMon2 Case

In this section we provide an specific analysis of the power consumption profiles drawn by the `idle`, `hdparm`, and `cpuburn` benchmarks when PowerMon2 is used to measure, independently, the 3.3 V, 5 V and 12 V lines on Intel_Desktop and AMD_Server. We also show how, by using an internal wattmeter like PowerMon2, it is possible to distinguish the power source line. The sum of all these lines is included in the results as well.

We depict the behavior of Intel_Desktop in Fig. 11. In this platform, depending on the benchmark, the different lines draw different powers. For `idle`, the 3.3 V and 5 V lines transport more power than the 12 V lines and power remains constant in all of them. The situation varies for `hdparm`: the 5 V lines fluctuate in conjunction with the 12 V socket-related lines; meanwhile the 3.3 V lines and 12 V mainboard lines show a fairly plain profile. The situation becomes even more interesting for the `cpuburn` benchmark. In this case we observe how

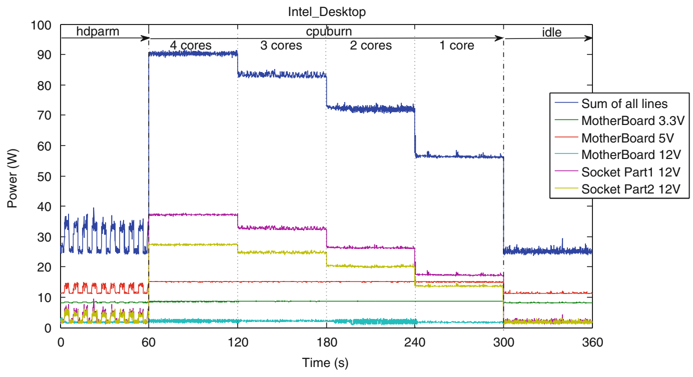


Fig. 11. Power consumption profile provided by PowerMon2 wattmeter when running different benchmarks on Intel_Desktop.

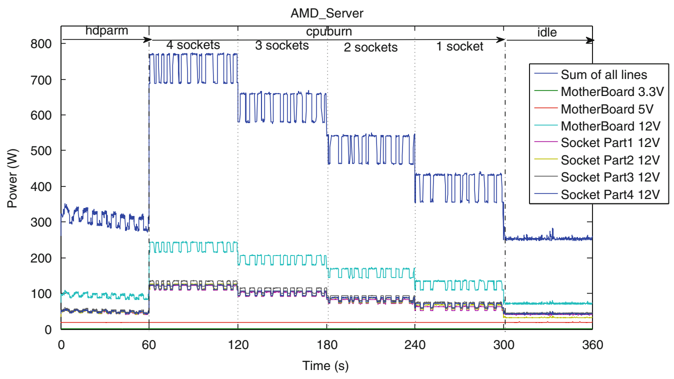


Fig. 12. Power consumption profile provided by PowerMon2 wattmeter when running different benchmarks on AMD_Server.

the 5 V and 12 V socket-related lines initially increase their power, which could be expected since the `cpuburn` processes highly stress the cores and, therefore the power dissipated by the cores. It is also important to note that the mainboard 3.3 V and 12 V lines feature a very plain profile.

The same experiment was repeated on AMD_Server and results are shown in Fig. 12. This platform exhibits a quite different range of lines that PowerMon2 is able to analyze. In this case we analyze the 3.3 V, 5 V and 12 V for mainboard and the 12 V socket-related lines. As shown in Fig. 12, each one of the 4 sockets of AMD_Server is not fed with one specific 12 V socket-related line. Indeed, while varying the number of sockets during the `cpuburn` benchmark, the power measured by the each of the four 12 V socket-related lines changes. The `idle` benchmark provides a flat profile for all the lines, nevertheless; the 12 V constantly transport more power than the 3.3 V and 5 V lines. For `hdparm` this situation changes, the 12 V lines start by drawing the natural spikes of this benchmark, specifically the 12 V mainboard line almost consumes twice the power of the 12 V socket-related lines. For the execution of the `cpuburn`, the situation is repeated, the 12 V mainboard line doubles the power drawn by the 12 V socket-related lines and starts dropping down when less sockets are working. For 3.3 V and 5 V lines is interesting to point out how plain the profile is.

6 Discussion and Conclusions

In this paper, we analyze and evaluate different external and internal power monitoring devices tested using two different computing systems, a server (AMD_Server) and a desktop machine (Intel_Desktop), offering a complete comparison in terms of power and energy consumption.

First of all, we show that, unlike external wattmeters, internal wattmeters do no register neither an equal energy consumption nor a similar power variability. Results show indeed that the energy and the power consumption captured by PowerMon2 are often higher than the ones provided by NI and DCM, especially for AMD_Server. These results can be explained by the fact that these wattmeters use different components to measure the power energizing the internal wires. As expected, we show that the energy consumption measured by the internal wattmeters is always lower than those measured by external ones. However, that was not always the case with the power measurements. As a matter of fact, when the `cpuburn` benchmark runs on AMD_Server, PowerMon2 registers higher power measurements compared to WattsUp, even if PowerMon2 does not take into account the PSU overhead and even some internal components. This result tends to show that PowerMon2 is less accurate when used to measure too high power consumptions (more than 500 W).

Contrary to what one could expect, we pointed out that the extra energy consumption (i.e. mean-idle) of a given benchmark running on a machine is not equal for all the wattmeters. Indeed, it signals a difference of more than 50% for benchmarks like `hdparm` and for low consuming ones line `iperf`. This generates doubts about the accuracy of the wattmeters. This paper pointed out that, contrary to the external measurements, the internal values are more dispersed and generate many outliers samples. This is certainly due to the high frequency of internal wattmeters. However the question that remains is whether this is because a very high frequency allows to measure some power fluctuations that the external wattmeters cannot provide or, instead, it is because it generates too much noise.

Unlike the power profiles obtained with the external wattmeters, the ones registered by the internal devices show some differences and a noise more visible with the NI. This behavior can be explained by the very high sampling rate of the NI. Furthermore, external wattmeters on the one hand and internal wattmeters on the other provide different power profiles and make them displaced for `hdparm` and, more specifically, `cpuburn`. These differences mainly come from differences built into the devices and potential environment changes like the room temperature. Contrary to Intel_Desktop, the power profile on AMD_Server permits to observe clear spikes and drops when running the `cpuburn` benchmark. This behavior is related to the numerous fans that are alternatively turned on and off in order to cool the server machine.

While plotting the power profiles with a varying sample rate, we highlighted that measuring at a very high sample rate (500 S/s) is not always necessary for profiling power dissipated by the applications, and may even provoke some noise that masks the general shape of the power trace. The appropriate sample rate is not always the highest possible but the one that best enables to understand the power fluctuations. Also, an internal wattmeter like PowerMon2 offers power profiles showing the different lines (3.3 V, 5 V and 12 V) in a separated way. This allows to detect where the power fluctuations come from. One may hope that each line is related to one specific component, which was not the case neither with Intel_Desktop nor AMD_Server.

In sum, thanks to the high sample rate and to the different measured lines, the internal measurement devices allow to better visualize some power fluctuations that the external wattmeters are not able to capture. However, a high sample rate is not always necessary to understand the evolution of the power consumption during the execution of a benchmark. A key to achieve accurate and reliable measurements requires a calibration with oscilloscope and previous tests which ensure their quality; however this operation is not always easy to perform. Moreover, monitoring very large-scale distributed systems (like exascale supercomputers) with internal wattmeters is not practicable, especially as these equipments are not easy to connect and the price per node is expensive (e.g., 2,700 € for NI). In this study, we focused on power measurement devices but other ways to measure the power consumption exist, like the IPMI (Intelligent Platform Management Interface) and RAPL (Running Average Power Limit) counters available on INTEL_DESKTOP. These internal sensors are promising techniques to measure power consumption for large scale distributed systems since they do not require an extra device. However, they are still not adapted since the power measurements they provide are not yet very accurate and their sample rate can be relatively low; moreover, they may introduce noise. From this paper, some open questions remain unsolved. Thus, as a follow-up to this work, we will further investigate to get the answers to some of the open questions that we expressed in the paper.

Acknowledgments. This research was supported by the European COST Actions IC804 ("Energy efficiency in large scale distributed systems") and IC805 ("Complex HPC systems"). Authors from Universitat Jaume I were also supported by project CICYT TIN2011-23283 and FEDER.

References

1. Hsu, C.H., Feng, W.C., Archuleta, J.S.: Towards efficient supercomputing: a quest for the right metric. In: Proceedings of the High Performance Power-Aware Computing, Workshop (2005)
2. Dongarra, J., et al.: The international ExaScale software project roadmap. Int. J. High Perform. Comput. Appl. **25**(1), 3–60 (2011)
3. Feng, W., Feng, X., Ge, R.: Green supercomputing comes of age. IT Prof. **10**(1), 17–23 (2008)
4. Laros III, J.H., Pedretti, K.T., Kelly, S.M., Shu, W., Vaughan, C.T.: Energy based performance tuning for large scale high performance computing systems. In: Proceedings of the Symposium on High Performance Computing, HPC '12, San Diego, CA, USA, pp. 6:1–6:10 (2012)
5. Alonso, P., Dolz, M.F., Igual, F.D., Mayo, R., Quintana-Ortí, E.S.: DVFS-control techniques for dense linear algebra operations on multi-core processors. Comput. Sci. - R&D **27**(4), 289–298 (2012)
6. Alonso, P., Dolz, M.F., Mayo, R., Quintana-Ortí, E.S.: Energy-efficient execution of dense linear algebra algorithms on multi-core processors. In: Cluster Computing, May 2012
7. Feng, W., Cameron, K.: The green500 list: encouraging sustainable supercomputing. Computer **40**(12), 50–55 (2007)

8. Ltaief, H., Luszczek, P., Dongarra, J.: Profiling high performance dense linear algebra algorithms on multicore architectures for power and energy efficiency. Comput. Sci. **27**(4), 277–287 (2012)
9. Subramaniam, B., Feng, W.: The green index: a metric for evaluating system-wide energy efficiency in HPC systems. In: 8th IEEE Workshop on High-Performance, Power-Aware Computing, Shanghai, China, May 2012
10. Ge, R., Feng, X., Song, S., Chang, H.C., Li, D., Cameron, K.W.: Powerpack: energy profiling and analysis of high-performance systems and applications. IEEE Trans. Parallel Distrib. Syst. **21**(5), 658–671 (2010)
11. Subramaniam, B., Feng, W.: Statistical power and performance modeling for optimizing the energy efficiency of scientific computing. In: Proceedings of the 2010 IEEE/ACM International Conference on Green Computing and Communications, GREENCOM, Washington, DC, USA, pp. 139–146. IEEE Computer Society (2010)
12. Alonso, P., Badia, R.M., Labarta, J., Barreda, M., Dolz, M.F., Mayo, R., Quintana-Ortí, E.S., Reyes, R.: Tools for power-energy modelling and analysis of parallel scientific applications. In: 41st International Conference on Parallel Processing - ICPP, pp. 420–429 (2012)

EnergyBox: A Trace-Driven Tool for Data Transmission Energy Consumption Studies

Ekhiotz Jon Vergara(✉) and Simin Nadjm-Tehrani

Department of Computer and Information Science, Linköping University, Linköping, Sweden
{ekhiotz.vergara, simin.nadjm-tehrani}@liu.se

Abstract. Although evolving mobile technologies bring millions of users closer to the vision of information anywhere-anytime, device battery depletions hamper the quality of experience to a great extent. We argue that the design of energy-efficient solutions starts by energy-awareness and propose EnergyBox, a tool that provides accurate and repeatable energy consumption studies for 3G and WiFi transmissions at the user end. We recognize that the energy consumption of data transmission is highly dependable on the traffic pattern, and provide the means for trace-based iterative packet-driven simulation to derive the operation states of wireless interfaces. The strength of EnergyBox is that it allows to modularly set the 3G network parameters specified at operator level, the adaptive power save mode mechanism for a WiFi device, and the different power levels of the operation states for different handheld devices. EnergyBox enables efficient energy consumption studies using real data, which complements the device-dependent laborious physical power measurements. Using real application transmission traces, we have validated EnergyBox showing an accuracy range of 94–99% for 3G and 93–99% for WiFi compared to the real measured energy consumption by a 3G modem and a smartphone with WiFi.

Keywords: Transmission energy · Trace-based simulation · Energy consumption studies · 3G · 802.11 · UMTS · WLAN

1 Introduction

Wireless interfaces account for a great energy cost on mobile devices [7,13], but unfortunately, there seems to be some misconceptions about how energy consumption is influenced by data transmissions. While some studies employ only bulk data transfer to measure energy consumption [4], they do not measure the impact of the applications' real data pattern on energy consumption. The data transmission pattern depends on the real application operation (influenced by application developers) and the user-device interaction [6,14], and drastically influences the energy per bit [3,17].

In this paper we argue that reducing the energy consumption of wireless transmissions begins by being aware of the energy consumption characteristics

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 19–34, 2013.
DOI: 10.1007/978-3-642-40517-4_2,

of different technologies. Since physical energy measurement studies are laborious and time-consuming, solutions are usually tested with little variation of factors such as network settings, hardware dependence, and synthetic data traces.

There are different approaches available to perform energy studies. While the professional testing equipment [1] or the physical power measurement equipment [19–21] provide high accuracy, they limit the studies to a fixed environment and incur high cost. Smart battery interfaces providing current values are available in some devices, and the software using them (e.g., CurrentWidget for Android or Nokia Energy Profiler) measures the aggregated current draw from all the device components. The error of the instant battery interface reading is usually high [5]. These and vendor specific development platforms [2] enable only device-dependent studies of aggregated consumption. We believe that a modular approach to carry out flexible and efficient energy studies isolating the transmission energy to complement physical energy measurements is essential.

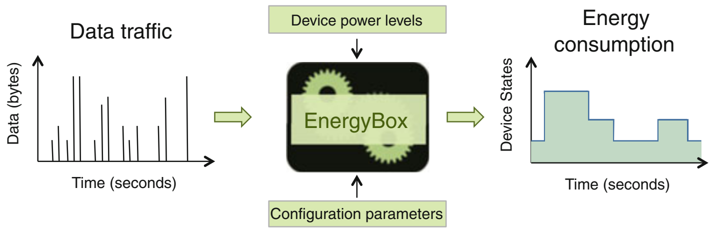


Fig. 1. Overview of EnergyBox function.

Our work proposes the design of EnergyBox: a tool that enables accurate studies of data transmission energy consumption at the user end, using real traffic traces as input. We focus our efforts on the most widespread wireless technologies (3G and WiFi) and capture application data transmission energy footprint at the user device. EnergyBox captures the underlying states of operation of the wireless interfaces. The hardware dependence is overcome by using parametrised device specific power levels. For a given data trace, EnergyBox automatically outputs the operation states over time, so that when combined with device specific power levels it enables the computation of energy consumption. The general idea of our EnergyBox is shown in Fig. 1.

The data traces can be directly captured on different devices or synthetically created in order to study the impact of different transmission mechanisms under different configurations for these wireless technologies. EnergyBox assists researchers and application developers to immediately estimate energy consumption of data transmissions for a diverse range of test cases. The scenario is set up using different data traces, device power levels and network settings.

The tool accepts the statically configured parameters at the radio layer such as inactivity timers, the data buffer thresholds and a low activity mechanism used by the operator for state transition decisions in 3G communication. For

WiFi, it incorporates the adaptive power save mode commonly used as the power saving mechanism in the latest generation devices. We validate our EnergyBox against physical measurements and show an accuracy range of 94–99% for 3G and 93–99% for WiFi compared to physically measured energy consumption. The EnergyBox has already been used in a research environment to facilitate energy consumption studies complementing physical measurements [22,23].

The rest of the paper is organised as follows: Sect. 2 provides the energy consumption background for 3G and WiFi. The design of EnergyBox is presented in Sect. 3 and this is evaluated against physical measurements in Sect. 4. Section 5 presents the related works and Sect. 6 covers concluding remarks and future work.

2 Background

We provide a background for communication energy footprint through illustrative measurements that show the main factors affecting the data transmission energy of 3G and WiFi at the device end. Figure 2 shows an overview of our measurement setup, which is also used for validation purposes in Sect. 4.

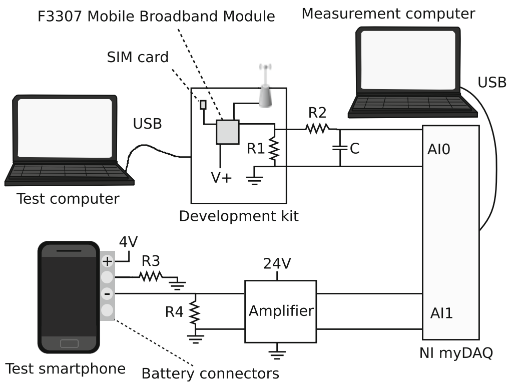


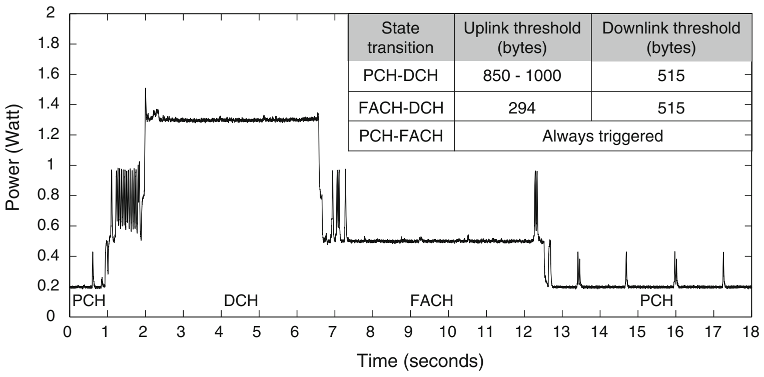
Fig. 2. Measurement setup for 3G and WiFi.

The 3G measurements were performed on a power-efficient mobile broadband module provided by Ericsson AB (KRY 901214/01, marked as Development Kit in 2). Since this exposes interfaces to measure the power consumption of the 3G modem easily isolating it from the rest of system (e.g., CPU or screen), we employ this platform instead of the 3G module in a smartphone. We measure the voltage drop over a shunt resistor ($R1=0.1\Omega$) and use a low-pass filter (approximately 16 Hz) to avoid any anti-aliasing effects [21].

The WiFi measurements were performed removing the battery of the smartphone under test, and adding a low-side sensing circuit ($R4 = 0.1\Omega$) with an isolating amplifier (maximum transmission error of 0.4%). We isolate the transmission energy from the rest of the system as in earlier works [19]. The power consumption is sampled at 1 kHz in both cases.

2.1 Energy Consumption of 3G

The energy consumption of the user equipment (UE) in 3G is mostly influenced by the radio resource management performed at the network operator side by the Radio Network Controller (RNC). The RNC uses the Radio Resource Control (RRC) and the Radio Link Control (RLC) protocols. According to the RRC, the UE implements a state machine where the states imply different performance (response time and data rate) and power consumption. The UE states are Dedicated Channel (DCH), Forward Access Channel (FACH), and Paging Channel (PCH), sorted from highest to lowest power drain. The transitions between states are controlled using inactivity timers and RLC data buffer thresholds.



State transition	Uplink threshold (bytes)	Downlink threshold (bytes)
PCH-DCH	850 - 1000	515
FACH-DCH	294	515
PCH-FACH	Always triggered	

Fig. 3. Example power profile for 3G and RLC buffer thresholds.

Figure 3 shows the power consumption levels of the states experienced at one location using the operator TeliaSonera in Sweden when sending a packet burst. In PCH, the UE can be paged with the lowest energy drain but no data can be sent. Some signalling is required to establish a connection and move from PCH to DCH before sending any data (1-2s in Fig. 3).

The UE reports to the RNC the observed traffic volume, that controls state transitions to higher performance states via the RLC protocol. The RNC employs fixed thresholds over the RLC buffer data occupancy and triggers the transitions when the thresholds are exceeded. The table in Fig. 3 shows the measured threshold values for the different state transitions.

Inactivity timers are used to move the UE to lower performance states. The UE moves to FACH after T_1 with small or no data transmission (7s in Fig. 3). T_2 controls the transitions from FACH to PCH. Inactivity timers create energy overheads known as energy *tails* due to the UE remaining in high consuming state while not transmitting anything [4,21].

To reduce these energy overheads, the Fast dormancy mechanism allows the UE to signal the RNC the desire to switch to the lowest power state by sending a Signalling Connection Release Indication before the inactivity timeout. Some networks implement a low activity mechanism in DCH to release the transport channel and move to FACH when there is low traffic [9]. Thus, we see that the energy tails and the above mechanisms make the energy consumption dependent on the traffic pattern and operator settings in a complex way.

2.2 Energy Consumption of WiFi

The transmission energy consumption for WiFi is mostly influenced at the driver level in the WiFi station (the client handset). The station is in the Constantly Awake Mode (CAM) when it has the power-saving features disabled experiencing the best performance.

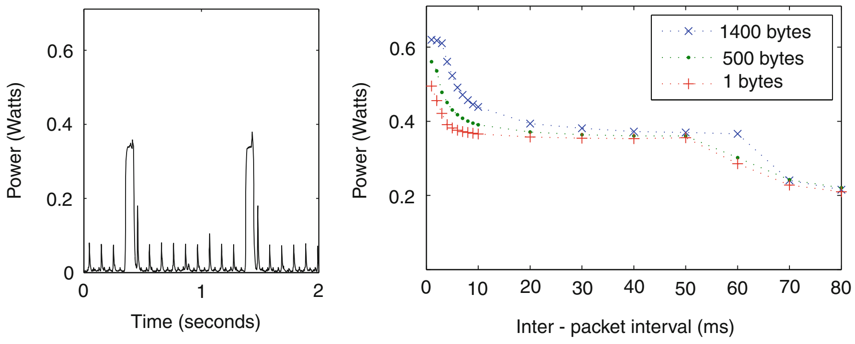


Fig. 4. Power levels of adaptive PSM for Xperia Arc.

The IEEE 802.11 standard defines a Power Save Mode (PSM), which allows the stations to switch to low power mode during predefined periods of time when not transferring any data. The access point (AP) buffers downlink frames for the clients and the clients wake up periodically (at multiples of the beacon interval). The clients send a Power Save Poll (PS-Poll) message to the AP to receive each buffered frame. Recent smartphones implement a mechanism named Adaptive PSM to overcome the overhead and latency drawback of using this PS-Poll mechanism [10]. The client switches between the CAM and PSM modes based on heuristics (e.g., number of packets, traffic inactivity period or screen on/off). The client uses the power management field in null data frames to inform the AP about its current mode.

Figure 4(left) shows an adaptive PSM implementation in a Sony Ericsson Xperia Arc smartphone, where the station moves to CAM for sending the traffic, and switches back from CAM to PSM after a predefined inactivity timeout

(δ) without packet transmission. This δ timeout creates an energy tail in the similar way as in 3G. Repeated measurements have shown δ to be around 220 and 70 ms for the Samsung Galaxy SII and Xperia Arc respectively, much shorter than in 3G. Previous generation devices implement longer δ timeouts (e.g., 1.5 seconds for HTC Magic) [11,16,20]. Moreover, some drivers also implement packets per second thresholds (*Up* and *Down*) that trigger PSM-CAM and CAM-PSM transitions respectively [15].

While in the same state, the station consumes more power when the data rate increases. In order to show the impact of data rate on transmission power and δ, we create an uplink data stream varying inter-packet interval and packet size using the Xperia Arc. Figure 4(right) shows that for higher data rate (i.e., short inter-packet interval), the average power level increases substantially. When the inter-packet interval is increased to 70 ms, the station switches back to PSM since the inter-packet interval is greater than δ. This drops the average power level. To sum up, adaptive PSM leads to a high impact by the data pattern on the energy consumption of a WiFi station.

3 EnergyBox

EnergyBox captures the influence of the data transmission pattern on transmission energy consumption by performing trace-based iterative packet-driven simulation. The usage of real data traces means that the corresponding energy is realistic, and reflects the impact of the data pattern on real throughput and delay in the network. EnergyBox also accepts using synthetic data traces as input, creating repeatable tests for a given purpose. Given a packet trace and configuration parameters, the EnergyBox outputs the device states $S(t)$ over time. The total energy consumption is calculated by associating these states with power levels and integrating them over time. Device-specific power level values can be obtained through measurement platforms as the one described in Sect. 2 or in the literature [4,16,19,21,24]. These operation power levels abstract the hardware dependency allowing to quantify an application footprint on a given device. EnergyBox simulates the 3G network parameters specified at operator level and the adaptive PSM mechanism specified at the handset driver for WiFi. The design of the EnergyBox is rooted in measurements and a careful literature study. By providing our code to the research and development community we offer a simple to configure but still powerful approach to perform accurate data transmission energy measurement studies.

3.1 3G Model

The RRC state transitions are captured by a parametrised finite state machine that simulate the inactivity timers, the RLC buffers and a low activity mechanism in a packet-driven manner. Figure 5a shows the states and the state transitions we use in our 3G model.

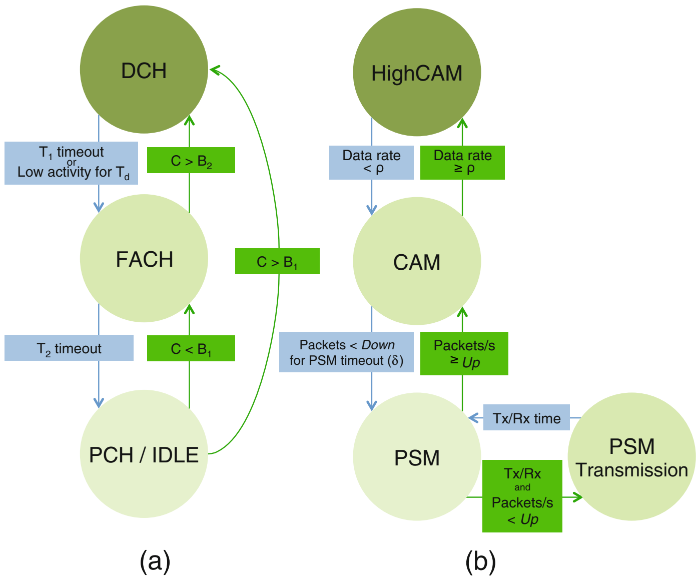


Fig. 5. Overview of the EnergyBox state machines for 3G (a) and WiFi (b).

For each packet P_i in the trace and its timestamp $t(P_i)$, we calculate $\Delta_i = t(P_i) - t(P_{i-1})$ as the elapsed time between the two packets. Δ_i is used to simulate the inactivity timers T_1 and T_2: if $\Delta_i > T_1$ or $\Delta_i > T_2$, we trigger the corresponding state transition. In order to account for the signalling time between states, every state transition has pre-defined a parametrised transition duration. This can be obtained from a simple power measurement (e.g., the delay observed in Fig. 3) or measuring the round-trip time for the different transitions.

There are four RLC buffer thresholds for PCH-FACH and FACH-DCH transitions (see Sect. 2): two uplink (B_1^u and B_2^u) and two downlink (B_1^d and B_2^d). The simulation of the RLC buffers is done as follows: we define Δ_i^u and Δ_i^d as the elapsed time from the last uplink or downlink packet (in the same direction). Note that since the size and direction (up/down) for each packet P_i is known, the current occupancy of each buffer in bytes can be computed. We denote this by C. The data is transmitted (i.e., the buffer is emptied setting $C = 0$) depending on the channel data rate. Given C and the channel data rate, the time to empty the RLC buffer can be calculated (T_e). T_e can be specified as a constant assuming constant data rate (e.g., $T_e = C/512$ kbps) or as function of buffer occupancy C based on data rate measurements in FACH for a network.

In order to simulate the low activity mechanism in DCH, we define T_d as the period of time over which the amount of data sent is monitored. We sum the size of packets during T_d and force the move to FACH when it is lower than a threshold. Fast dormancy is modelled by simply moving to PCH/Idle after a predefined time.

The current 3G model does not fully consider the impact of the received signal strength (RSS). Under poor radio-link conditions the transmission power is higher and the data rate is lower [21]. The data rate is captured by the recorded input data trace. However, in order to model the impact of RSS-power level we would have to feed the EnergyBox with a trace of received signal strength or similar indicator (e.g., Signal-to-Noise Ratio). This can be added to the implementation if the means of capturing such traces are available.

The strength of modelling 3G states in EnergyBox is that it is a generic way of capturing different state machines for different operators using a parametrised finite state machine: if an operator implements a RRC state machine where a single packet triggers a state transition from Idle/PCH to DCH, we only have to set the B_1^u threshold to 0 bytes.

3.2 WiFi Model

The EnergyBox WiFi captures the adaptive PSM mechanisms based on the inactivity timer and the number of packets per second. Handsets switch between two states (PSM and CAM) using adaptive PSM, but in order to model the high data rate behaviour of a handset, we define the state machine shown in Fig. 5b.

The station only wakes up for beacons in the *PSM* state. *PSM Transmission* state represents the sending or reception of packets in the PSM mode, where the station switches to a high power only during the transmission interval. The Tx/Rx time is defined as reconfigurable parameter, which is obtained by sending few packets and observing the power profile (similar to Fig. 4(left)).

In *PSM*, the transition to *CAM* is triggered whenever the number of packets per second count is higher than the *Up* threshold. When the number of packets per second is less than *Down* for a predefined timeout time (δ), the station switches back to *PSM*.

We have observed that the power drain of the station increases with the throughput as it was illustrated in Sect. 2. This behaviour is captured by computing the throughput over a short time window and comparing it to a ρ threshold. Once the time window has been set, the ρ threshold can be decided by sending a train of packets with different inter-packet interval (similar to Fig. 4(right)) and measuring the power drain. When the threshold is exceeded, the station switches to *HighCAM*, a state with a higher power level than *CAM*.

4 Evaluation

This section describes the evaluation of the EnergyBox accuracy against physical energy measurements using the setup described in Sect. 2. We define two metrics to quantify the accuracy of the EnergyBox: *energy accuracy* and *time accuracy*, which are described together with the methodology. The general methodology is similar for both WiFi and 3G and follows the following steps: (1) A set of applications are used to create different traffic patterns. We simultaneously collect packet traces and measure the *power trace* $P_d(t)$ of the device while creating traffic from one application at a time. This is done using the toolset described in Fig. 2. The power trace $P_d(t)$ represents the ground truth. (2) From the measured power trace, we compute the expected *device states* $S_d(t)$ using power level thresholds. These thresholds are obtained earlier using measurements on a given handset. (3) The EnergyBox is fed with the packet traces and outputs the *inferred states* $S_i(t)$. (4) We compare the EnergyBox inferred states $S_i(t)$ against the device states $S_d(t)$ and compute the difference over time. T represents the total duration of a trace and $f(x, y)$ simply returns 1 if $x = y$, and 0 otherwise. Time accuracy represents the percentage of time that the inferred states and the measured states overlap (i.e. the higher the measure the better the accuracy over a time interval). It is defined as follows:

$$Time\ Accuracy = \frac{\int_0^T f(S_i(t), S_d(t))\,\mathrm{d}t}{T} \cdot 100\ (\%) \tag{1}$$

(5) The EnergyBox assigns the measured device-specific handset power levels for each state to the inferred states obtaining the *inferred power trace* $P_i(t)$. The energy consumption is computed in Joules integrating $P_i(t)$ over the trace duration (T). Energy accuracy reflects the difference between the inferred energy consumption and the energy consumption of the measured power trace (the ground truth):

$$Energy\ Accuracy = \frac{\int_0^T P_i(t)}{\int_0^T P_d(t)} \cdot 100\ (\%) \tag{2}$$

4.1 Methodology and Evaluation Settings

The packet traces are 5 minutes long, captured with *tcpdump* for WiFi and 3G. We demonstrate the reliability of the EnergyBox by covering a wide range of data patterns in terms of inter-packet interval, packet size and total amount of data transmitted created by commonly used mobile applications. Since the EnergyBox is deterministic (i.e., it creates the same output for a given input packet trace and the configuration parameters), each packet trace is fed only once in the EnergyBox. We employ 10 different packet traces coming from different applications for 3G and 9 for WiFi. Email has periodic small data transmissions, whereas Facebook and Web represent bursty downloads of bigger amounts of data. Spotify is a music streaming application that sends data in different bursts and Stream is a constant radio stream. Skype Chat represents instant messaging services which usually have smaller data transmissions. Skype Call and Video are audio and video conferences with some small chat messages. Finally, Youtube captures the user watching videos online. As opposed to doing several experiments per application, we chose to test several different applications with one trace each, similar to the approach in other current work [5,13,17].

For 3G, we set $T_1 = 4.1$ s and $T_2 = 5.6$ s and the RLC buffer thresholds shown in Fig. 3. The state transition times are 1.7, 0.43, 0.65 s for PCH-DCH, PCH-FACH and FACH-DCH respectively. We set the time to empty the RLC buffers, $T_e = 1.2 \cdot C + 10$ as a function of the buffer occupancy (C) based on repeated data rate measurements in FACH following a previous methodology [17]. We enable the low activity mechanism (with $T_d = 4\,\text{s}$ and B_1^u as the threshold) and disable fast dormancy since the 3G module does not support it. The device-specific power levels used are the average power values measured on the 3G module for different states: DCH, FACH and PCH (1.3, 0.5 and 0.2 W). The evaluation was performed under similar and typical values of received signal strength.

For WiFi, we set $Up = 1$ and $Down = 1$, which are the settings of the stations used in our measurements. The Galaxy SII was used for the evaluation and δ was set to 220 ms based on observation (similar to Fig. 4(left)). We used 30, 250 and 500 mW for PSM, CAM and HighCAM respectively based on measured average power levels. The ρ data rate threshold was empirically set to 3 kB for a time window of 50 ms based on empirical experiments mentioned in Sect. 2.2. Since adaptive PSM in the Galaxy SII differentiates only two states (PSM and CAM), we perform the time accuracy evaluation of the EnergyBox WiFi using the two basic states of our WiFi state machine.

4.2 EnergyBox 3G

The average accuracy range of EnergyBox over the different traces for 3G is high, 95% and 98% for time and energy respectively. Figure 6 shows the time and energy accuracy for the different traces.

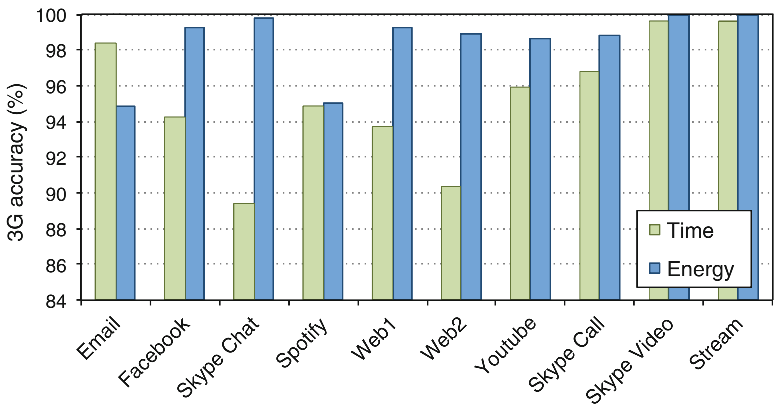


Fig. 6. EnergyBox 3G accuracy for different traces.

Regarding lower time accuracies (e.g., Skype Chat), the typical cause is that the inferred state is DCH state while the real state is FACH. The source of this deviation is the time to empty the buffers (T_e) of the RLC buffer simulation. Even though the modelling of this parameter is based on real measurements, the variations in number of active users and traffic at the operator network makes the real time to empty the buffers different from the measurement-based model in some instances. The Skype Chat trace has many small packets. These stress the RLC data buffer simulation. However, the accuracy is still close to 90%.

The energy accuracy is high in all the traces. The lowest registered energy accuracy is for Email and Spotify traces, which is still higher than 94%. Figure 7 shows an example of the EnergyBox accuracy by comparing it to the real measurements for the Email trace.

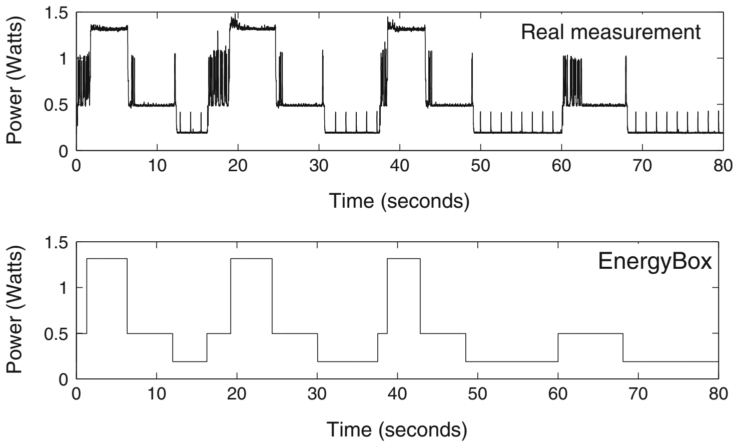


Fig. 7. A fragment of a 3G measurement and EnergyBox 3G inferred output for the Email trace.

4.3 EnergyBox WiFi

The average accuracy of EnergyBox over the different traces for WiFi is really high with 99% and 98% for time and energy respectively. Figure 8 shows the accuracy for the different traces. Concerning time accuracy, the few discrepancies are originated from the value given to δ, which makes the switch from CAM to PSM somewhat earlier than in the real states.

Regarding the energy accuracy metric, the differences can be traced to traces with higher data rates (e.g., Skype Video and Stream). The simplicity of the fixed power level for *HighCAM* in our model does not completely cover the case of high data rate with a higher power level than the fixed one assigned to *HighCAM*. However, the discrepancy is still relatively small.

To summarise, the above comparisons provide evidence that EnergyBox provides high energy accuracy and device state information when using real traces.

5 Related Works

We categorise the main body of related work into two areas: energy consumption measurement studies and the characterisation of energy consumption of wireless networking at the user end.

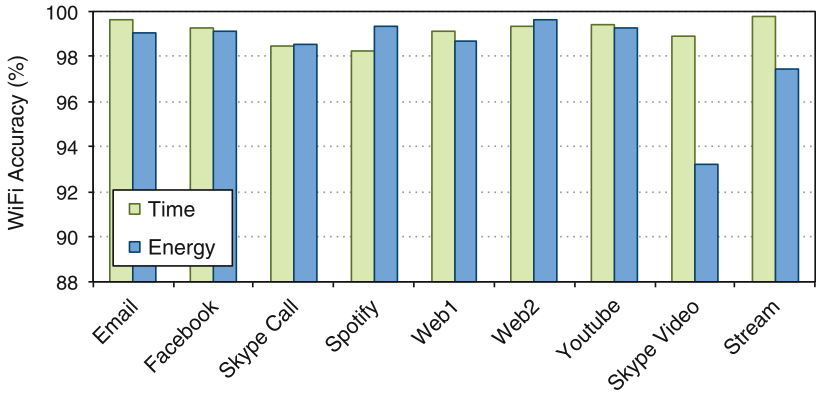


Fig. 8. EnergyBox WiFi accuracy for different traces.

Energy measurement studies: an influential measurement study by Balasubramanian et al. [4] reports the categorisation of three different energy components in cellular networks: ramp, transfer and tail. The tail is the most consuming component and it is caused by the inactivity timers statically set by the network operators. Our previous works [3,21] refine this study by performing physical measurements using a cellular modem that isolates the data transfer energy.

Qian et al. [18] perform a detailed study of the tail energy overhead using data traces retrieved from a network operator and later [17] point out how different applications inefficiently utilise the radio resources due to their data pattern.

The main advantage of our 3G measurement setup is that we are able to *isolate* the energy consumption for *data transfers* by validating the transmission-specific measurements in a 3G module, whereas earlier works do not isolate the energy for data transfers from the rest of the system when evaluating accuracy.

Wang et al. [24] present the energy consumption of data transfers for different packet sizes and transmission intervals over WLAN and cellular networks. Rice et al. [19] measure the WiFi energy consumption in a variety of smartphones. Several works focus in studying the energy consumption of WiFi based on bulk data transmissions [4,7]. Studies using bulk data transmissions do not capture the impact of adaptive PSM on energy consumption. Various works [11,16,20] study the different adaptive PSM mechanisms in previous generation devices. A recent study by Pyles et al. [15] shows the *Up* threshold for different devices.

The main advantage of our validation base is the fine-grained measurements isolating data transfer energy. Moreover, most studies do not consider the impact of the data traffic pattern on energy consumption and focus on bulk transfers.

Wireless energy consumption characterisation: Balasubramanian et al. [4] model the energy consumption of 3G and WiFi using linear regression based on their bulk data download measurements. Their simple model does not capture the impact of the data pattern on the energy consumption. Oliveira et al. [12] use a state based model for 3G modelling the RLC buffer occupancy as fixed data rate (kbps) without performing any validation of their model.

Some works [26,27] analytically model different aspects of the energy consumption of 3G. Our approach is applicable to arbitrary data traces captured from any real application, whereas named works are limited to modelling web and streaming traffic.

The radio resource usage application profiler for 3G presented by Qian et al. [17] is the work closest to our EnergyBox for 3G. Their focus is on providing the application developer insights and hints about how to reduce the energy consumption due to 3G transmissions. We extend their 3G work by including in our 3G state-based model a low activity mechanism implemented by some operators, thereby improving the accuracy of the model. Moreover, our EnergyBox captures different RRC state machines within a single parametrised one.

The work by Harjula et al. [8] creates a device-specific power profile for different messaging intervals based on device measurements on a Nokia 95 for a single operator. Their model uses indicators such as average packet size and signalling frequency that need to be extracted from the data transmissions. Instead, EnergyBox directly works on data traces and derives the RRC states of different networks simulating the RLC buffer thresholds and state transitions. Our approach is more general and simplifies energy consumption studies.

Xiao et al. [25] present a detailed power level model for the 802.11g interface. The modelled data pattern based on traffic burstiness was only validated by simulating TCP download and upload traffic based on the amount of data transferred. In comparison, our adaptive PSM model is validated against a set of representative application traces. Dong et al. [5] propose to automatically self-generate a system energy model within a device (including WiFi and 3G) using the smart battery interface.

A recent study by Pathak et al. [13] present a system-call-based power modelling for smartphones, including device dependent 3G and WiFi models. Tracing system calls typically requires modifications to the operating system (e.g., root access), whereas we adopt a generic approach based on packet captures, which are generally available[1].

6 Conclusion and Future Work

Designing energy efficient solutions for wireless networks starts by energy awareness. To reduce energy waste, there is a need to provide visibility and understanding of the energy consumption that impacts battery lifetime at the user end as well as elsewhere in the network. Our contribution makes the energy footprint of the network interface explicit, and presents a tool that provides accurate energy consumption values given packet traces as input. We focus the EnergyBox on the most widespread wireless interfaces (3G and WiFi) and capture the parameters influencing the energy consumption most.

[1] Android phones provide traffic statistics that can be captured every millisecond and packet level capture is available without root permission from Android 4.0 on.

The design of EnergyBox is rooted in our physical energy measurements as well as a careful literature study. We build on the fact that the handset energy consumption in 3G is essentially driven by the inactivity timers and the RLC data buffer thresholds that control RRC state transitions at the network operator end. The energy consumption in WiFi is mostly influenced by the handset dependent adaptive PSM mechanism implemented at driver level in the handset. We capture these aspects within parametrised state machines in the EnergyBox and evaluate its accuracy against physical measurements. Our evaluation shows that the EnergyBox provides accurate energy consumption estimates.

Including the impact of the signal strength into EnergyBox and extending it towards other wireless interfaces such as the fourth generation cellular network (LTE) and Bluetooth would be interesting directions for future work. In its current state, however, the EnergyBox is found to be a valuable instrument in studying the energy consumption in different networking scenarios in our current research. It has substantially aided efficient energy-related studies by emulating different parameter setups and replacing energy measurements.

Acknowledgment. This work was supported by the Swedish national graduate school in computer science (CUGS). The authors wish to thank the support of Ericsson AB, and in particular B-O Hertz, Pär Emanuelsson and Claes Alströmer for providing the 3G developer kit and facilitating the measurement gathering phase.

References

1. T&m solution. rohde & schwarz. http://www.rohde-schwarz.com/en/applications/optimize-the-quality-of-experience-of-mobile-devices-application-card_56279-35727.html
2. Trepn profiler, qualcomm. https://developer.qualcomm.com/mobile-development/development-devices/trepn-profiler
3. Asplund, M., Thomasson, A., Vergara, E.J., Nadjm-Tehrani, S.: Software-related energy footprint of a wireless broadband module. In: Proceedings of the 9th ACM International Symposium on Mobility Management and Wireless Access, MobiWac '11. ACM (2011)
4. Balasubramanian, N., Balasubramanian, A., Venkataramani, A.: Energy consumption in mobile phones: a measurement study and implications for network applications. In: Proceedings of ACM Internet Measurement Conference IMC (2009)
5. Dong, M., Zhong, L.: Self-constructive high-rate system energy modeling for battery-powered mobile systems. In: MobiSys '11, pp. 335–348. ACM (2011)
6. Falaki, H., Mahajan, R., Kandula, S., Lymberopoulos, D., Govindan, R., Estrin, D.: Diversity in smartphone usage. In: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services, MobiSys '10, pp. 179–194. ACM (2010)
7. Friedman, R., Kogan, A., Krivolapov, Y.: On power and throughput tradeoffs of wifi and bluetooth in smartphones. In: Proceedings of IEEE INFOCOM 2011, pp. 900–908 (2011)
8. Harjula, E., Kassinen, O., Ylianttila, M.: Energy consumption model for mobile devices in 3g and wlan networks. In: IEEE Consumer Communications and Networking Conference (CCNC 2012), pp. 532–537 (2012)

9. Holma, H., Toskala, A.: WCDMA for UMTS: HSPA evolution and LTE. Wiley Online Library: Books, John Wiley & Sons (2010)
10. Krashinsky, R., Balakrishnan, H.: Minimizing energy for wireless web access with bounded slowdown. In: Proceedings of the 8th Annual International Conference on Mobile Computing and Networking, MobiCom '02, pp. 119–130. ACM (2002)
11. Manweiler, J., Roy Choudhury, R.: Avoiding the rush hours: Wifi energy management via traffic isolation. IEEE Trans. Mob. Comput. **11**(5), 739–752 (2012)
12. Oliveira, T., Ursini, E., Timoteo, V.: Simulation inspired model for energy consumption in 3g always-on mobiles. In: IEEE 2nd National Conference on Telecommunications (CONATEL 2011), pp. 1–7 (2011)
13. Pathak, A., Hu, Y.C., Zhang, M.: Where is the energy spent inside my app?: fine grained energy accounting on smartphones with eprof. In: Proceedings of the 7th ACM European Conference on Computer Systems, EuroSys '12, pp. 29–42 (2012)
14. Paul, U., Subramanian, A., Buddhikot, M., Das, S.: Understanding traffic dynamics in cellular data networks. In: Proceedings of IEEE INFOCOM 2011, pp. 882–890 (2011)
15. Pyles, A.J., Qi, X., Zhou, G., Keally, M., Liu, X.: Sapsm: smart adaptive 802.11 psm for smartphones. In: Proceedings of the 14th International Conference on Ubiquitous Computing, UbiComp '12. ACM (2012)
16. Pyles, A.J., Ren, Z., Zhou, G., Liu, X.: Sifi: exploiting voip silence for wifi energy savings in smart phones. In: Proceedings of the 13th International Conference on Ubiquitous Computing, UbiComp '11, pp. 325–334. ACM (2011)
17. Qian, F., Wang, Z., Gerber, A., Mao, Z., Sen, S., Spatscheck, O.: Profiling resource usage for mobile applications: a cross-layer approach. In: MobiSys '11, pp. 321–334. ACM (2011)
18. Qian, F., Wang, Z., Gerber, A., Mao, Z.M., Sen, S., Spatscheck, O.: Characterizing radio resource allocation for 3g networks. In: Proceedings of the 10th Annual Conference on Internet Measurement, IMC '10, pp. 137–150. ACM (2010)
19. Rice, A., Hay, S.: Measuring mobile phone energy consumption for 802.11 wireless networking. Pervasive Mob. Comput. **6**, 593–606 (2010)
20. Rozner, E., Navda, V., Ramjee, R., Rayanchu, S.: Napman: network-assisted power management for wifi devices. In: MobiSys '10, pp. 91–106. ACM (2010)
21. Vergara, E.J., Nadjm-Tehrani, S.: Energy-aware cross-layer burst buffering for wireless communication. In: Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet, e-Energy '12. ACM (2012)
22. Vergara, E.J., Prihodko, M., Nadjm-Tehrani, S.: Mobile location sharing: an energy consumption study (poster). In: Proceedings of the 4th International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet, e-Energy '13. ACM (2013)
23. Vergara, E.J., Sanjuan, J., Nadjm-Tehrani, S.: Kernel level energy-efficient 3g background traffic shaper for android smartphones. In: Proceedings of the 9th International Wireless Communications and Mobile Computing Conference (IWCMC 2013) (2013)
24. Wang, L., Manner, J.: Energy consumption analysis of wlan, 2g and 3g interfaces. In: Proceedings of the 2010 IEEE/ACM International Conference on Green Computing and Communications & International Conference on Cyber, Physical and Social Computing, GREENCOM-CPSCOM '10, pp. 300–307. IEEE Computer Society (2010)

25. Xiao, Y., Savolainen, P., Karppanen, A., Siekkinen, M., Ylä-Jääski, A.: Practical power modeling of data transmission over 802.11g for wireless applications. In: Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking, e-Energy '10, pp. 75–84. ACM (2010)
26. Yang, S.R., Yan, S.Y., Hung, H.N.: Modeling umts power saving with bursty packet data traffic. IEEE Trans. Mob. Comput. **6**(12), 1398–1409 (2007)
27. Yeh, J.H., Chen, J.C., Lee, C.C.: Comparative analysis of energy-saving techniques in 3g pp and 3gpp 2 systems. IEEE Trans. Veh. Technol. **58**(1), 432–448 (2009)

Myths in PMC-Based Power Estimation

Jason Mair(✉), Zhiyi Huang, David Eyers, and Haibo Zhang

Department of Computer Science, University of Otago, Dunedin, New Zealand
{jkmair, hzy, dme, haibo}@cs.otago.ac.nz

Abstract. Many techniques have previously been proposed for using low-level CPU Performance Monitoring Counters in power estimation models. In this paper, we present some common myths of these techniques, and their potential impact. Such myths include: (1) sampling rate can be ignored; (2) thermal effects are neutral; and (3) memory events correlate well with power. We aim to raise the awareness of these interesting issues, which existing power modeling techniques usually do not address. Our discussions provide some guidance to avoid these myths and their effects through detailed specification of software and hardware configurations.

Keywords: Power estimation · Performance Monitoring Counter (PMC) · Power metering model

1 Introduction

Much previous research has focused on building accurate power estimation models based upon Performance Monitoring Counters (PMCs) [8,10–14]. These techniques allow runtime power estimation, on a per-application basis, without requiring the use of a power meter or special hardware. They are very useful in power metering of virtual machines in cloud computing [15,16]. Even hardware-based power metering in Intel's Sandy Bridge microarchitecture has to use PMC-based estimation to measure the dynamic power of the cores [9]. However, the research into the use of these techniques needs to be coupled with an improvement in the specification of the hardware and software configurations used in the process of the researchers' power modeling. In previous research, these configuration details were normally given a brief mention, but lacked discussion and analysis on their potential impacts on the resulting power model. Our experiments show that many of these configuration details are crucial in order to achieve reproducible results. The omissions that we have encountered can often be attributed to reasons such as assumption of background knowledge, limited space for publication, or indifference to the authors interpretation of the experimental results.

Detailed specification of the hardware and software configurations used in the process of power modeling is also crucial for the avoidance of myths in this research area. It can help prevent the readers from drawing erroneous conclusions.

For example, suppose there are two different power models with an identical mean measuring error of 5%, but for different machines with an equal total power

DOI: 10.1007/978-3-642-40517-4_3,

consumption of 200 W. Without further information, the natural conclusion that can be drawn by the reader is both power models are equally accurate. However, a different conclusion can be drawn if it was additionally known that machine 'A' has a dynamic power (aka workload dependent power) of 30%, while machine 'B' has a larger dynamic power of 60%. Given a total error of 10 W (5% of total 200 W) and the constant static power (aka base power), it is now clear that the model based on machine 'A' has a 16.6% error for dynamic power while the model based on machine 'B' has a much lower error of 8.3% for dynamic power. The extra information on the proportion of static power and dynamic power enables a fair comparison to be made between the two, otherwise identical, power estimation models.

In this paper, we discuss the impact of hardware and software configurations and the possible myths in the research area of PMC-based power estimation due to the lack of detailed information. More precisely, we have identified the following three possible myths.

– *Sampling rate can be ignored*—In previous research, the sampling rate for taking PMC-power samples is seldom mentioned. Very often, when a benchmark is run, the PMC counters are recorded for the whole execution period of the benchmark and the average power of the benchmark during the period is used in the sample. However, we will show that the sampling rate can affect the accuracy of the power estimation.
– *Thermal effects are neutral*—Thermal effects are often not discussed in power estimation. Many experimental results were given without mentioning the temperature condition of the CPU chips. We will show that the temperature of the CPU chips has an effect on the accuracy of the power model and its effect can be eliminated with proper treatment.
– *Memory events correlate well with power*—Previous research often assumes a correlation between memory events and power. Intuitively memory events like cache misses should correlate with power consumption due to many memory fetches. We will show that this is not the case on our multicore system possibly due to a different memory architecture. This observation gives us inspiration that detailed specification of software and hardware configurations is very important for fair comparison and deep understanding of different PMC-based power models.

The intention of this paper is not to criticize any of the existing work, or to propose any alternative methodologies. Instead, our objectives are twofold. First, we would like to raise awareness of the potential pitfalls in the research area. Second, we want to stress the importance of detailed specification of software and hardware configurations in the process of power modeling. If experimental results are published with sufficient context and specification, it helps avoid the myths that we will discuss in power modeling.

The remainder of this paper is organized as follows. Section 2 describes our experimental setup and software/hardware configuration. The myths are discussed in Sect. 3. Related work is in Sect. 4, with the conclusions in Sect. 5.

2 System Configuration

In this section and throughout this paper, we make a conscious effort to specify all relevant configuration information, allowing experimental results to be placed in the appropriate context. It will be shown through this paper that changes in sampling rate, execution time and configuration of benchmarks impact the resulting model. If such information was not explicitly specified, as is often the case in many published papers, it becomes harder to compare fairly among alternative approaches and power models. A commonly omitted data value in a power estimation model is the amount of static vs. dynamic power within a system. Static power is the constant, workload independent power consumption of components like the Power Supply Unit (PSU). Dynamic power changes according to the workload, with the most obvious example of the CPU.

Care should be taken to ensure that any neglected details do not impact on the resulting power model or create the potential for erroneous conclusions. Often researchers neglect the detailed specification of system parameters due to limited publication space. Unfortunately, a large number of parameters can have a significant effect on the experimental results. The architecture-specific nature of PMC-based power estimation means it is important to document all relevant hardware and software configurations.

It is for these reasons that this section takes the time to describe the overall experimental setup of hardware and software configuration, including the design of our own micro-benchmark. Certain experiment-specific configuration details are left until the corresponding results sections.

2.1 Experimental Setup

The experiments are run on a Dell PowerEdge R905 with four quad-core AMD Opteron 8380 processors (CPUs), with each core having its own floating-point unit (FPU). Each processor is located with 4GiB of memory organized in a NUMA (Non-Uniform Memory Accesses) architecture, providing a total of 16GiB RAM. The processor provides four alternate operating frequencies through DVFS (Dynamic Voltage and Frequency Scaling), however we restrict the frequency to the highest (2.5 GHz) for our experiments as it is the most commonly used frequency. All benchmarks are compiled using gcc 4.6.3 with no optimization enabled, ensuring none of the micro-benchmark operations are optimized away. OpenMP 3.0 [5] is used for the NAS Parallel Benchmarks, which are compiled with gcc 4.6.3, using optimization argument -O3. All benchmarks were running on a standard installation of Linux version 2.6.32-25.

The power is measured with the Watts Up? PRO .net power meter, connected via USB to an external monitoring system. The accuracy of the power meter is $\pm 1.5\% + 0.3$ watts [1]. An iSocket (InSnergy Socket)[1] power meter was

[1] Institute for Information Industry, http://web.iii.org.tw/, who we thank for providing this measurement equipment.

additionally used to validate power measurements, which has an accuracy of 1%. The measured base/idle power (i.e., static power) for our server is 249 W.

The monitoring system was additionally configured to remotely monitor the server's system components and temperatures, while recording the power. This was achieved by altering an instance of IPMItool [7] to log specific measurements at the same rate as the power values. Communication is handled by the Intelligent Platform Management Interface (IPMI) over LAN, bypassing the OS, directly communicating with the Baseboard Management Controller (BMC). Thus, this measurement causes no overhead on the test server.

All performance values are collected from the PMCs, which are a set of four, per-core hardware registers [6]. Each register is set to record one of the 120 available performance measures at the start of each execution.

2.2 Benchmark Configuration

The myths discussed in this paper come from observations made during our work on deriving a runtime power estimation model based on PMC values. The idea behind such a model is to find the relationships between key PMCs, the workload type they represent and the resulting power use. For example, if the PMC for FPU utilization is high, the current workload is FPU intensive, causing a large power draw. Alternatively, if the number of cache misses is high, there may be many memory accesses, resulting in lower utilization of the processor and a lower power draw from the processor.

To explore these basic principles, we created a micro-benchmark designed to reproduce five key workload types that could be expected within a system, shown in Table 1. The micro-benchmark, as shown in Program 1, consists of a larger outer loop, intended to perform a large number of iterations, and a small `for` loop for short bursts of each workload type. The inner `for` loops execute quickly, ensuring that whenever the PMCs are sampled during execution, a mix of workload types will be represented. Modest variations in this burst time are provided by the pseudo random numbers. This is then multiplied by a ratio, which is designed to allow extra weight to be added to a single workload type during execution.

Table 1. Types of workload

Micro-benchmark	**Description**
FPU	Floating point multiplication
INT	Integer multiplication and division. Represents most micro operations
memory	Random memory accesses
NOP	idle loop with NOPs
cache	Memory accesses with high cache-hit ratio

By default, all ratios are set to one. However, if we are interested in the impact of a more FPU-oriented workload, we increase the FPU ratio, leaving all other ratios the same. The ratios shown throughout this work are 1, 2, 4, 6 and 8. To avoid any synchronization overheads, 16 independent concurrent instances are run across all 16 cores.

```
for(large_number_of_iterations)
  for(pseudo_random_number x fpu_ratio)
    fpu_micro();
  for(pseudo_random_number x int_ratio)
    int_micro();
  for(pseudo_random_number x memory_ratio)
    memory_micro();
  for(pseudo_random_number x nop_ratio)
    nop_micro();
  for(pseudo_random_number x cache_ratio)
    cache_micro();
```

Program 1: Pseudo-code of the micro-benchmark

3 Myths

Many alternative models have been proposed for PMC-based power estimation [8,10–14]. They commonly used a black-box approach to data processing, where the regression function is directly applied to the data without necessarily being visualized. In contrast, this section describes some of the observations made during our thorough analysis and visualization of the collected experimental data in the process of power estimation modeling. We will also disclose the differences of the resulting model due to the system architecture used and the selection of statistical methods.

Not all of our observations have gone without a note in the literature, but often only warrant a passing mention without much discussion or analysis. This may be due to space limitations. Also it is worth noting that failing to disclose all relevant information would not affect the correctness of the previous work.

In this section, each subsection introduces a myth that could be observed in published literature. Each myth is followed by a series of observations in our experiments and our approach to avoid the myth. Each myth is then concluded with a brief discussion of how this should be taken into account within our modeling process.

3.1 Myth 1: Sampling Rate can be Ignored

The rate at which PMCs and power samples are taken can have a direct impact on the strength of correlation and the noise within a dataset. This is intuitive, as it will determine the aggregation of results, but is often not documented.

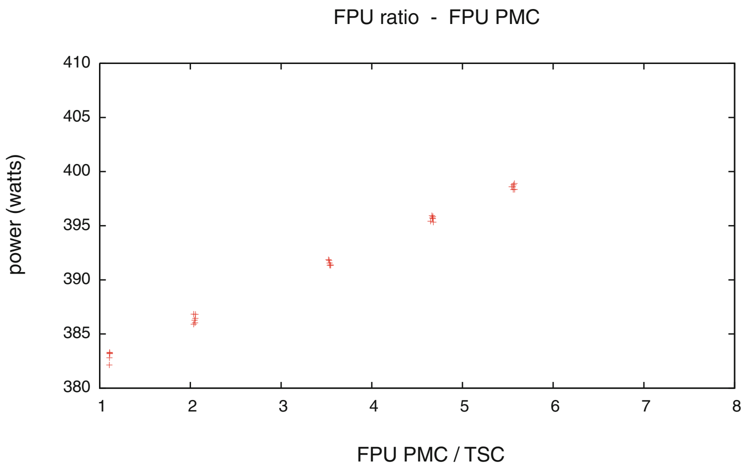


Fig. 1. Intensity of FPU correlated with power measured over entire execution of micro-benchmark running an FPU workload.

Observation: execution time and sampling rate The first step in deriving the power model will consist of collecting some initial coarse-grained data for a range of PMCs, sampled once when execution begins and again upon completion of each micro-benchmark iteration. The TSC (Time Stamp Counter) is additionally read whenever PMCs are read in order to provide a measure of time. TSC measures the time as the number of cycles since reset, allowing the intensity of the PMC-related event to be calculated during execution.

For example, Fig. 1 shows some intensity values for the FPU activities for multiple iterations of the micro-benchmark, introduced in Sect. 2.2, in different configurations. The x-axis is the intensity of FPU activities, which is calculated by taking the difference in the two FPU_{PMC} measurements of the execution period and dividing it by the difference in the corresponding TSC values, which provides the intensity of the activities during the execution period. The y-axis is the calculated average power use, measured by the power meter, over the same execution period. In this example, the FPU ratio is adjusted for different iterations, providing the spread of data clusters along the x and y-axis.

In Fig. 1, the data points form a series of tight clusters along a linear path, which is what was expected according to previous research.

Knowing this works at the most basic level, the logical next step is to increase the rate at which samples are taken. This was chosen to be once every second, in order to match that of the power meter. Each PMC value and power measurement are logged to a file during execution, with a corresponding timestamp, to allow synchronization and post-processing. This time, the results shown in Fig. 2 were not what was expected. The modest horizontal spread of points within each cluster is due to the adjustments in the pseudo random number in the algorithm. However, the vertical stripping was not expected at all. This indicates that some-

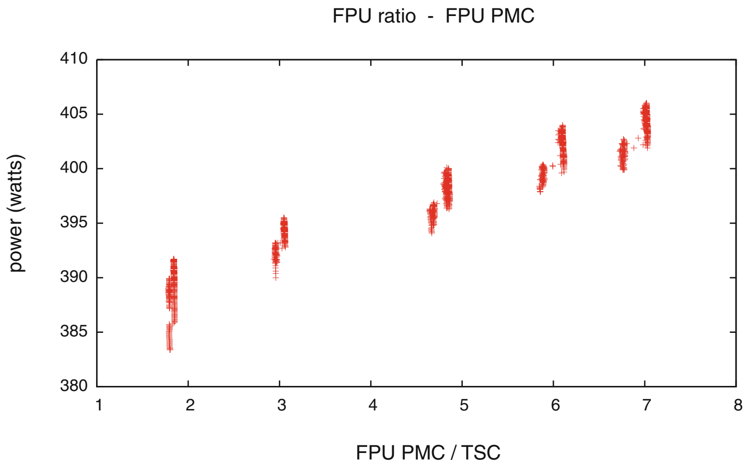


Fig. 2. Intensity of FPU correlated with power, sampled every second for ∼3 minute execution of the micro-benchmark running FPU workload. Vertical stripes are a result of a warm-up effect.

thing within the system is causing the power values to vary during execution, which was previously obscured by the coarse-grained samples.

Since the micro-benchmark consists of a fairly consistent workload, it was suspected that the stripping was due to an inherent latency in the system responding to starting execution. If such a latency exists, the trend in the data should become more linear with increased execution times. Therefore, each configuration was re-run, increasing the execution time from ∼ 3 minutes to ∼ 25 minutes. The results in Fig. 3 show that when sampled over a longer period with the same sampling rate, data points return to lying on a linear path, though there is a long vertical tail in each cluster, which will be explained in the next section. The spread of points for each cluster along the x-axis is a result of the increased range of pseudo random times made available by the significantly increased iteration count.

In this section we have seen two ways in which important data may be obscured. First, the extreme case of using a sampling rate which is too coarse-grained for the benchmark being sampled. Second, execution times which are too short will hide longer run trends within a dataset.

Observation: benchmark configuration and execution time The previous observation not only illustrated the importance of choosing an appropriate rate at which to sample a benchmark, but also the importance of ensuring a sufficiently long execution time. However, care must be taken not to introduce more noise when increasing execution time. While the micro-benchmarks experience minimal workload variation during execution, this will not be the case for all other benchmarks. Therefore, increasing execution time in the same way will not have the same effect, contributing more noise than expected.

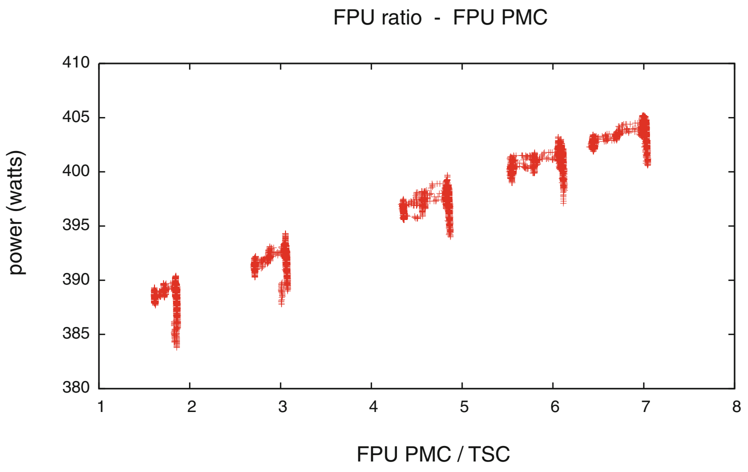


Fig. 3. Intensity of FPU correlated with power, sampled every second for ~25 minute execution of the micro-benchmark running FPU workload. The linear trend is more clear though vertical stripes still exist due to a warm-up effect.

To test this hypothesis, we ran an OpenMP instance of the Fast Fourier Transform (FFT) benchmark from the NAS Parallel Benchmark (NPB) suite in two different configurations. To ensure each instance to experience the same latency from startup, a 10 minute idle period is run before each instance begins. The first configuration had a problem size of 512 $\times$ 512 $\times$ 512, completing 250 iterations. The mean power measured was 403.6 W with a standard deviation of 56.19 W. Looking at the data we found many small periods of low power use during the execution due to the large number of iterations performed, explaining the large deviation in measured power.

For comparison purposes, an alternative configuration was run with a problem size of 1024 $\times$ 512 $\times$ 512, completing 100 iterations. The mean power of 425.0 W was much closer to the peak power, with a smaller standard deviation, 36.2 W.

From this observation, we find it is important to carefully consider how benchmarks are configured to avoid introducing any unexpected noise in measured power.

Discussion When selecting the rate at which to sample the PMC counters, it is important to consider potential smoothing effects if samples are too coarse grained. Not doing so will obscure data trends and characteristics for the sampled benchmark. Similar to this is the importance of execution time, as this additionally impacts the number of samples for a given rate, potentially further obscuring long-run trends.

Documenting both the sampling rate and the execution time help to add context to the commonly reported statistics like the mean power. The significance of such values can be questionable and statistically inaccurate if the data set is believed to be too small.

Also it is good to give the standard deviation of the measured power so that the smoothness of the power changes is known. Mean power value can hide the smoothness of the power trend, as two power trends with the same mean value can have very different standard deviations. Standard deviation can be used to reflect how reliably the mean power value is used to characterize the power feature of the execution period.

3.2 Myth 2: Thermal Effects are Neutral

Thermal effects are often not discussed in power modeling. For those who are aware of the thermal effects on power consumption, it is commonly perceived that the thermal effects can be negated by locking the fan speed, believing the change of fan speed is the main cause of the variation in power due to changes in thermal load. An alternative technique to locking the fan speed is the use of a CPU warm-up phase before the start of each execution. However, we find neither of these can sufficiently negate the thermal effects.

Observation: fan speed and power Section 3.1 mentioned the presence of a long-run latency in power changes corresponding to the beginning of each micro-benchmark execution, which causes the vertical long tail of each cluster in Fig. 3. The most likely cause of delayed effect on power within a system is temperature related. To explore this, we used IPMI to monitor CPU temperatures once a second during execution of each micro-benchmark. Figure 4 shows the normalized values for power and temperature of all four CPUs on the y-axis. The x-axis gives

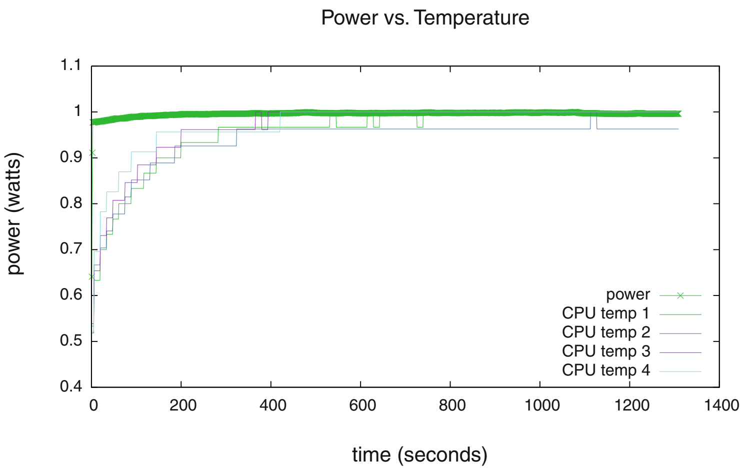


Fig. 4. Normalized CPU temperatures and power meter readings for a long-run execution of the micro-benchmark running FPU workload.

the time in seconds. The power curve steadily increases until around 400 seconds where it flattens out. The recorded temperatures follow a similar trend where they continue to increase until about 400 seconds, reaching a stable point with the exception of an occasional temperature spike.

These results illustrate a trend between CPU warm-up and the corresponding power latency. The most surprising aspect of this is the length of time required to reach a stable value, 400 seconds. In many cases this will be longer than the execution time of the benchmarks.

To further confirm this relationship, IPMI was used to monitor the fan speed for each CPU. Despite the temperatures changing, the fans' speed remains constant at 3600rpm. This speed is even maintained under a high thermal load running a CPUburn benchmark [4]. Contrary to the common belief, the power latency is not caused by changes in the speed of the fans when they respond to an increased thermal load. Therefore, policies designed to lock fan speeds through the BIOS are not capable of negating all of the dynamic thermal effects on power consumption.

Observation: warm-up and cool-down In an attempt to remove the effect of the warm-up phase, the micro-benchmark was re-run after a CPU warm-up phase. There is a 15-minute cool-down period before each iteration of the micro-benchmark starts to execute, in order to ensure consistent starting temperature for each iteration. After that, an instance of the CPUburn was run on each core for different execution times, providing different times of CPU warm-up. The results for two different CPU warm-up lengths of 60 and 90 seconds are shown in Figs. 5 and 6, respectively. Same as the previous figures, the x-axis is

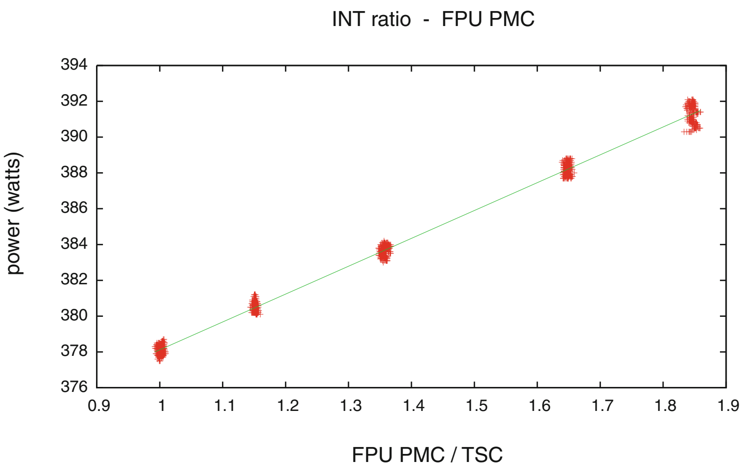


Fig. 5. Intensity of FPU correlated with power, sampled every second for the micro-benchmark running INT workload. Sampling starts after a 60-second period of CPU warm-up.

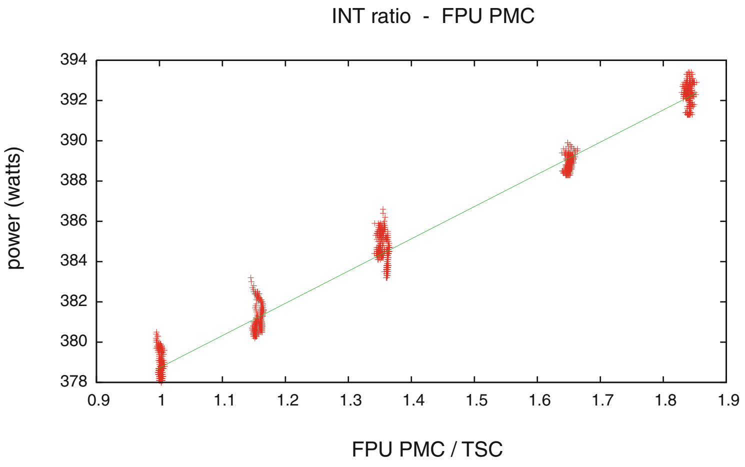


Fig. 6. Intensity of FPU correlated with power, sampled every second for the micro-benchmark running INT workload. Sampling starts after a 90-second period of CPU warm-up.

the intensity of the FPU activities, while the y-axis is the power in Watts. This time, the INT ratio is adjusted between iterations, giving the spread of clusters along the x-axis.

A CPU warm-up phase of 60 seconds, as shown in Fig. 5, is enough to eliminate much of the vertical tail caused by the warm-up phase, resulting in a stronger linear correlation. Alternatively, with the CPU warm-up phase of 90 seconds, as shown in Fig. 6, it begins to over-warm the CPU, which causes the opposite effect, a vertical stripping above the main linear trend, instead of the vertical tail.

However, different types of workload in the benchmark need different warm-up periods. A 60-second CPU warm-up phase provides the best results for a workload with lots of integer calculations, as shown in Fig. 5. However, a workload with lots of floating point calculations requires a 90-second CPU warm-up. These results are illustrated by the Pearson's Correlation Coefficient for the two workload types in Table 2. While differences in correlations are not significant they do illustrate the point that no single CPU warm-up policy is sufficient for all workload types.

Discussion Temperature variations within a system have the potential to adversely impact the accuracy of a power estimation model. For example, an instance of the micro-benchmark running a FPU workload initially uses 397 W, stabilizing at around 405 W. This gives an error of 2% of total power, and more significantly, 5% of dynamic power. Due to the myth of the thermal effects, there is no single solution designed to mitigate all thermal effects.

Table 2. Pearson's Correlations for different workload type with different warm-up times

CPU warm-up time (seconds)	FPU-type	INT-type
30	0.969611	0.977467
60	0.992794	0.997928
90	0.996615	0.992321
120	0.993053	0.985091

Unfortunately, it is not likely that a single policy exists to reliably remove all warm-up effects on power consumption. The most likely cause for the warm-up effects is static power leakage from the processor, which is due to the high temperatures. For example, a 12% reduction in CPU (dynamic) power was made in [2] by reducing the operating temperature, while maintaining the same voltage and frequency 4.6GHz.

It might seem that the only way to reliably monitor thermal effects is through embedded temperature sensors. However, since their placement inside the socket is some distance away from the top of CPU, embedded sensors do not provide reliable temperature data [3]. Also such sensors were not designed for high precision temperature reading, as their purpose is to provide an early warning system to prevent hardware damage.

In summary, we have made two key observations. First, benchmarks can experience a large warm-up effect on power consumption during their start-up, which is not due to the changes in fans speed. Second, the length of warm-up phase required varies between different workloads, as illustrated through differences in the warm-up times required by FPU and INT workloads. That means using a fixed period of warm-up for all workloads will not achieve the desired result.

3.3 Myth 3: Memory Events Correlate Well with Power

The correlation between memory-related PMCs like cache miss and memory activities is intuitive and well known. However, there is a myth that memory-related PMCs correlate equally well with power consumption. This proved not to be true for our multicore system.

Observation: neutrality of memory-related PMCs Memory-bound and CPU-bound workloads exhibit quite different power characteristics and in most cases are therefore treated differently. For power estimation, it is also common to use those PMCs with a direct logical connection to memory use. This intuitively makes sense, and is what we expected to be the case too. In modeling the power use of memory workloads, we used PMCs directly related to memory, such as, instruction cache miss, data cache miss, L2-cache miss, L3-cache miss, DTLB miss and DRAM accesses.

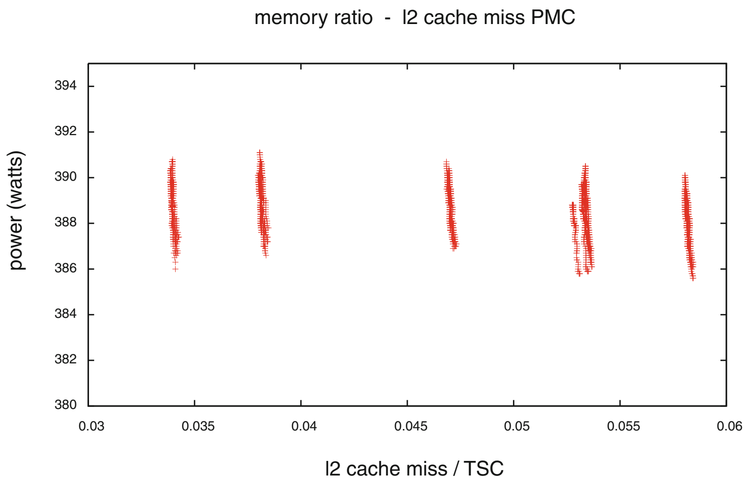


Fig. 7. Intensity of L2-cache misses correlated with power, sampled every second for the micro-benchmark running memory workload.

Surprisingly, none of these counters provided a strong correlation between memory use and power consumption. To illustrate this, the results for L2-cache misses are plotted in Fig. 7. Our micro-benchmark has been configured to execute a large number of memory accesses by increasing the ratio for memory accesses, leaving all other micro-benchmark ratios at the default value of one. The x-axis is the intensity of cache misses, which is calculated taking the difference of two L2-cache-miss PMC values divided by the elapsed TSC value. The y-axis is the measured power. We collect the intensity and power samples at every one second. The warm-up effect, seen by the vertical stripping within each cluster, is present since there is no CPU warm-up phase before data collection.

The most notable observation to be made in Fig. 7 is the distinct lack of any vertical offset between clusters. It seems that power is not functionally determined by L2-cache misses. The same results have been found for all other memory-related PMCs mentioned, so we do not repeatedly show the results here.

Discussion A common approach taken in building a power model is to decompose the processor and the expected workload type into several key components, such as FPU, Memory, Stalls, and Instructions Retired [8]. Each component requires a specific, strongly correlated PMC to represent its power consumption. In the case of memory, a PMC like cache misses is expected to correlate well with the activities of the memory subsystem. This approach has worked in other power estimation models [12], but failed to do so on our experimental system due to the neutrality of memory-related PMCs to power consumption.

This difference of results can likely be attributed to the architectural differences, though we are not sure which components have caused the difference, as there are several components which could possibly contribute to such differences. The first component could be the memory architecture. Our system uses NUMA, where, unlike some systems of the previous work, there is no single memory bank shared between all processors. The memory in our system is arranged in 4GiB blocks beside each of the four processors. Given the random memory accesses, extra overhead may be incurred if memory accesses are shifted to a remote processor's memory block.

Also the processors in our system lack the ability to sleep, even at low levels. That means the processor maintains a busy loop or executes some other work while waiting for requests from the memory subsystem. Given the high thermal latencies, temperatures will remain high, despite lower levels of utilization.

4 Related Work

Much of the prior work on PMC-based power estimation has taken the approach of using PMCs to model the underlying architectural components, which are applied in a variety of use cases. Singh et al. [8] proposed a model which used micro-architectural knowledge to decompose the processor into its four main functional units: FP Units, Memory, Stalls, and Instructions Retired. PMC selection is made from initial data collected from the execution of the SPEC benchmark suite. A separate micro-benchmark is designed for each of the four PMCs most strongly correlated with power for each functional unit. The micro-benchmark data is used to form a piece-wise linear function.

Bertran et al. [12] take an even finer grained approach by starting with a set of about 97 micro-benchmarks designed to individually highlight all possible power components. This results in multiple linear equations with an input for each of the seven derived power components. During the runtime period, PMC multiplexing is required, as the micro-architecture does not allow that many counters to be collected simultaneously.

Da Costa et al. [14] present a methodology intended to broaden the range of modeled workloads by supplementing PMC values with process and system level statistics. This means the resulting model, derived through multivariate regression, is not limited to estimating the power of CPU and memory workloads, allowing accurate power estimation of the network and disk synthetic benchmarks.

Such models derive a single, global power estimation function typically used for monitoring system power on a per application basis. Alternatively, Alonso et al. [11] takes this more targeted approach in proposing a framework for instrumenting source code functions with power metering. The API logs PMC data and power values to derive a specific power model offline, enabling execution traces of power to be used during runtime estimation.

Wang et al. [10] takes the novel approach of using the fewest PMCs possible. The model was built using only CPU operating frequency and IPC, making it universal across microarchitectures. It is built into the SPAN libraries and interfaces to provide source code power estimation.

Dhiman et al. [16] presented a model for accurate power estimation in a virtualized environment. A performance counter manager is run on each host machine, designed to collect and correlate PMC events to each VM. Power estimates are then periodically made for each VM using classification based Gaussian mixture models.

Only a select sample of previous work is presented here to demonstrate some alternate uses for PMC-based power estimation models. A more comprehensive survey on hardware, software and hybrid power estimation techniques can be found in [10]. Given this varied use of PMC-based power models, each myth within this paper was presented and discussed without explicit guidance, ensuring all conclusions remain relevant and universally applicable to different approaches.

5 Conclusions

In this paper we have presented and discussed three myths in PMC-based power estimation models: sampling rate can be ignored; thermal effects are neutral; and memory events correlate well with power. The truth of each myth is revealed through a series of observations made while deriving our own PMC-based estimation model.

Such myths have arisen due to a lack of configuration specifications and accompanying analysis in published literature. This is of particular importance for PMC-based power estimation as the models derived are largely architecture dependent. As we have seen, changes in hardware and software configurations can adversely impact the resulting power models.

While failing to disclose all relevant information would not affect the correctness of the previous work, it can lead to erroneous conclusions being drawn from the results.In raising awareness of the impact of the missing relevant information, we hope researchers in this community more readily document hardware and software configurations in the future. Doing so will prove to be beneficial to the advancement of the research community.

Acknowledgement. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

References

1. Watts up? operators manual. https://www.wattsupmeters.com/secure/downloads/manual_rev_9_corded0812.pdf
2. Anandtech, Overclocking CPU/GPU/Memory Stability Testing Guidelines. http://forums.anandtech.com/showthread.php?p=34255681
3. Overclockers, Reconciling CPU Temperature Measures. http://www.overclockers.com/reconciling-cpu-temperature-measures/
4. Ubuntu Manuals, CPUburn. http://manpages.ubuntu.com/manpages/precise/man1/cpuburn.1.html

5. O.A.R. Board, OpenMP Application Program Interface Version 3.0, May 2008
6. AMD. BIOS and Kernel Developer's Guide (BKDG) For AMD Family 10h Processors (2009)
7. IPMItool. http://ipmitool.sourceforge.net/
8. Singh, K., Bhadauria, M., McKee, S.A.: Real time power estimation and thread scheduling via performance counters. SIGARCH Comput. Architect. News **37**(2), 46–55 (2008)
9. Rotem, E., Naveh, A., Rajwan, D., Ananthakrishnan, A., Weissmann, E.: Power Management Architecture of the 2nd Generation Intel Core microarchitecture, formerly codenamed Sandy Bridge. In: Hot Chips: A Symposium on High Performance Chips, August 2011
10. Wang, S., Chen, H., Shi, W.: SPAN: a software power analyzer for multicore computer systems. Sustain. Comput. Inf. Syst. **1**(1), 23–34 (2011)
11. Alonso, P., Badia, R.M., Labarta, J., Barreda, M., Dolz, M.F., Mayo, R., Quintana-Orti, E.S., Reyes, R.: Tools for power and energy analysis of parallel scientific applications. In: Proceedings of International Conference on Parallel Processing (ICPP), September 2012
12. Bertran, R., Gonzlez, M., Martorell, X., Navarro, N., Ayguade, E.: Decomposable and responsive power models for multicore processors using performance counters. In: Proceedings of the 24th ACM International Conference on Supercomputing, ICS 2010, Tsukuba, Ibaraki, Japan, pp. 147–158. ACM (2010)
13. Chen, X., Xu, C., Dick R., Mao, Z.: Performance and power modeling in a multi-programmed multi-core environment. In: Proceedings of the 47th Design Automation Conference, pp. 813–818. ACM (2010)
14. Da Costa, G., Hlavacs, H.: Methodology of measurement for energy consumption of applications. In: 2010 11th IEEE/ACM International Conference on Grid Computing (GRID). IEEE (2010)
15. Kansal, A., Zhao, F., Liu, J., Kothari, N., Bhattacharya, A.A.: Virtual machine power metering and provisioning. In: Proceedings of the 1st ACM Symposium on Cloud, Computing, pp. 39–50 (2010)
16. Dhiman, G., Mihic, K., Rosing, T.: A system for online power prediction in virtualized environments using Gaussian mixture models. In: Proceedings of the 47th ACM IEEE Design Automation Conference, pp. 807–812. ACM (2010)

Energy Consumption Library

Leandro F. Cupertino(✉), Georges Da Costa, Amal Sayah, and Jean-Marc Pierson

Toulouse Institute of Computer Science Research (IRIT), University of Toulouse III (Paul Sabatier), Toulouse, France
{fontoura, dacosta, sayah, pierson}@irit.fr

Abstract. The energy consumption of a computing system depends not only on its architecture, but also on its usage. This paper describes the Energy Consumption Library (`libec`), a modular library of sensors and power estimators, which do not depend on wattmeter to measure the power dissipated by a machine and/or the applications that it executes, etc. In addition, four use cases are used to demonstrate some of the library's capabilities.

1 Introduction

The power dissipated on data centers is highly increasing along time. It is known that the cost of maintaining such servers during two years can be greater than the cost of the hardware itself. The energy fraction spent by Information and Communications Technologies (ICT) over the worldwide available electricity is estimated to double in twelve years [1].

Initially the focus of energy savings on ICT was related to hardware enhancements. However, the power dissipated by a computing system is not static, i.e., it depends not only on its hardware specification but also on its usage. Distinct workloads will waste different amount of energy, which can vary even for the same application running on the same hardware depending, for instance, on its communication issues. The understanding of how the energy is used by an application can be used in software engineering to deploy libraries, implementations or compilation parameters to achieve energy-aware software. This knowledge can also be used on data centers to schedule the resources properly, taking into account the available electrical energy contracts. Several papers proposes different power models for estimating the energy consumption of applications according to its workload [2–5].

In this paper we present the Energy Consumption Library (`libec`), an open source library of sensors that can estimate the power dissipated by a machine or an application even without the presence of a wattmeter on the host machine. The remainder of this paper is divided as follows. Section 2 describes the library and its features. In Sect. 3 we present two use cases for this library, a process monitor and an application profiler. Finally, Sect. 4 draws some conclusions over the developed library.

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 51–57, 2013.
DOI: 10.1007/978-3-642-40517-4_4,

2 The Library

The main goal of the Energy Consumption Library, `libec`, is to provide a modular library to aid the development of new power estimators. To be easy to extend and maintain, it was implemented in C++ and is distributed under the GNU General Public License (GPL) version 3.0 or later. It can be downloaded from [6]. This library contains a set of sensors as input variables in several power models. The information provided from the sensors comes mainly from Linux kernel's API, `/sys` and `/proc` file systems. Nevertheless, these sensors can be extended in order to collect different data coming from any source specific sensors.

The `libec` sensors can be implemented at two levels: machine and/or application. The application level contains all the sensors that can be directly associated with a process identification (PID), these sensors are mainly related to software usage, such as performance counters. Meanwhile, the machine level has not only the aggregated value for all the processes, but also some physical properties measurements that cannot be associated to a PID, such as the CPU thermal dissipation. Furthermore, there is a special kind of application level sensor which is application's power estimators. The next subsections describe each available sensor.

2.1 Application/Machine Level Sensors

In order to estimate the energy consumed by an application, one need to have at least one application related variable, i.e., a variable which can retrieve application's information. With that in mind, `libec` has some PID related sensors. These sensors can gather not only application, but also machine level information.

The most power consuming devices on servers are CPU, memory and disk. In other words, in order to achieve good power models, one needs to access information regarding the usage of such devices. On the CPU side, one can exploit Performance Counters, CPU time, CPU elapsed time and CPU usage. Furthermore, sensors with memory usage and disk read/write can be used for memory and disk modeling, respectively. The available application's sensors for are implemented as follows.

Performance Counters (PCs) are hardware event counters that use special file descriptors to count them. They are available through the Linux kernel API [7]. This sensor gives the count hits between two updates, which are defined by the user. PCs can provide information related to a CPU, such as clock cycles, instructions, cache references and misses, branch instructions and misses, page faults, context switches, among others. In order to access the performance counters, one needs to have administrative privileges.

In order to enable more flexibility, the *CPU usage* sensor is composed by two other intermediate sensors: *CPU time* and *Elapsed CPU time. CPU time* provides the total CPU time, i.e., the sum of the system and user times. This information is retrieved from the `/proc/[PID]/stat` file. For the machine level information, this data is available in the `/proc/stat` file. This sensor can also provide the total elapsed time between updades, i.e., system, user and idle time. The returned time value is provided in clock ticks. The *Elapsed CPU time* sensor uses the information from the *CPU time* sensor to measure the CPU time difference between two updates. *CPU usage* (CPU%) provides the percentage of CPU utilization of a specific PID or CPU core. For the moment it cannot return the utilization of both at the same time, i.e., one cannot request the CPU usage of a PID regarding to one specific core, but only the PID's CPU usage on the entire machine or the core's usage for the machine as a whole. This sensor uses the elapsed CPU time sensor and divides it by the machine level CPU elapsed time, i.e., the total elapsed time.

Memory usage (MEM%) provides the percentage of memory used by a given process. It collects application's resident set size and divide it by the total available memory found in the `/proc/[PID]/stat` and `/proc/meminfo` files.

Disk Read/Write provides the number of read/written bytes between function calls available at the `/proc/[PID]/io` and `/sys/block/[dev]/stat` file. This sensor can retrieve information for any file partition that may come from a flash drives or an IDE hard drive.

2.2 Machine Level Sensors

In addition to the application level sensors, which can also be used to collect machine related information, `libec` contains some other sensors that can only be attributed to the machine as a whole. This section describes the sensors that are only available at the machine level.

The *CPU temperature* sensor retrieves its information from the `/sys` file system, but its file varies according to the CPU vendor. This information is triggered at the constructor of the class.

CPU frequency is also available in the `/sys` file system. In order to read such file, one needs to install the required packages and have administrative privileges. To enable all users to use such sensor, even if the information access is slower, the `/proc/cpuinfo` file is used according to the user's privileges.

The *Networking* traffic information is retrieved from the `/proc/net/dev` file. The user must define if the retrieved data will come from the sum of the networking devices or from just one of them. Besides, user can decide which type of data will be used (packets/bytes received/transmitted).

Some wattmeters interfaces can also be found on libec. To facilitate the library's use not only on server but also on notebooks, there is a meter which collects PDU power directly and another one which exploits the ACPI information to estimate the portable device power. The *ACPI Power Meter* retrieves information related to the voltage and current drained by the battery from the `/sys/class/power_supply` folder and calculates its power consumption. Its drawback is that it requires ACPI enabled hardware. On the server side, the PDU's communication deeply depends on the vendor's protocol used. Due to personal uses, we made available meters for some of the Grid5000's experimental testbed nodes [8], the RECS system [9] and Energy Optimizers Limited's Plogg (an outlet adapter to measure the dissipated power of devices).

2.3 Application's Power Estimators

The main target of this library is to enable users to develop new power estimators. For the moment, one static and two dynamic model where implemented. Static models require a priori information, while dynamic ones can auto adapt to different workloads but must have access to a power meter.

The simplest static model is a CPU proportional model. Our static model, namely *CPU MinMax*, is a linear estimator based on the minimum (P_{min}) and maximum (P_{max}) power consumption of the machine. This information must be provided by the user. It uses a *CPU usage* sensor to weight the power variance and the number of active processes ($|AP|$) to normalize the idle power (P_{min}) for each process as stated in Eq. 1. One must be aware that this estimator is architecture dependent and its performance varies according to the accuracy of the data provided by the user.

$$P_{pid} = (P_{max} - P_{min}) \times CPU\%_{pid} + \frac{P_{min}}{|AP|} \quad (1)$$

When a wattmeter is available, one can exploit it to achieve more precise results or to calibrate their models to use in similar machines that do not have such device. The *Inverse CPU* power estimator uses the information from the total energy consumption of a machine and attributes it to the application level by the use of a *CPU usage* sensor as stated below

$$P_{pid} = P_{machine} \times CPU\%_{pid}. \quad (2)$$

One well known method for achieving dynamic estimators is the use of linear regression method to weight some pre-defined sensors and find a model without user provided information. The *Linear Regression Dynamic Power Estimator* can do so by estimating the weights (w_i) for any application level sensor (s_i) within the follow equation

$$P_{machine} = w_0 + \sum_{i=1}^{n} w_i \times s_i. \quad (3)$$

3 Use Cases

In this section, we present four use cases for the `eclib`. The idea here is not to evaluate the results in a detailed way, but only to show the user, what kind of information it can produce using `libec`. The first use case shows how to implement your own sensor for it to be compatible with the other tools developed with `libec`. The second illustrates an easy way to monitor the top consuming processes running on the machine through the `ectop` tool. The third one is how to profile the energy spend by an application with Valgreen [10]. Finally the importance of a workload adaptative model is illustrated by changing the load of a machine and monitoring different types of power estimators. All of these use cases are available within the `ectools` package [6].

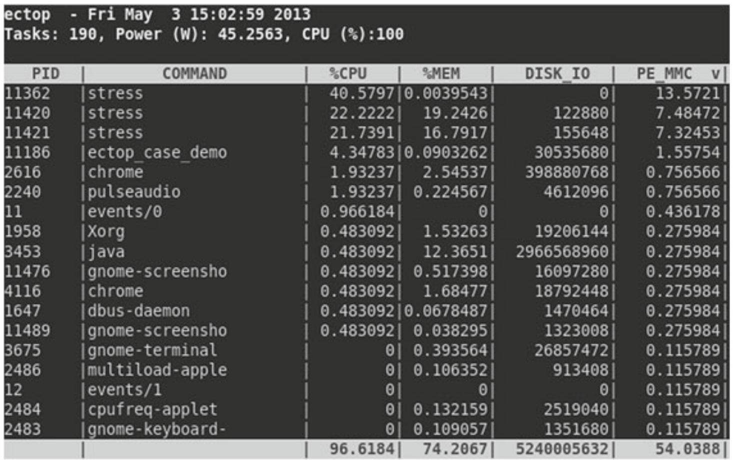
3.1 Extending Sensors

New machine and application sensors can be easily implemented by extending the *Sensor* and *SensorPID* classes, respectively. To do so, the user must at least overload the *update* and *updatePid* methods, if specific data structures are used, it may be necessary to overload the *getValue* and *getValuePid* methods as well.

3.2 Power Monitoring Tool

The Energy Consumption Monitoring Tool (`ectop`) application was conceived to provide an easy way to monitor power estimators and compare their accuracy in real time. It is a command line interface in which the user can keep track of the top consuming processes. It also allows the user to add/remove application level sensors and power estimators. To add/remove a sensor from the interface, one simply needs to instantiate the desired sensor class and add it to the monitor though the *addSensor* method.

Figure 1 shows an example of `ectop` with three sensors (CPU and memory usage and disk I/O) and one power estimator (min-max CPU proportional, PE_MMC). The `ectop` monitor shows a sum bar for each sensor's column, which gives an idea of its system wide values. For the presented scenario the sum of the power estimations is bigger than the value given in the power field. The power field is filled with the ACPI power estimator. This difference occurs because of the quality of the power estimator used. As stated earlier, `eclib` is a library to aid the development of new estimators and, for the moment, the power estimators present on this library are first attempts to generate broader models.



```
ectop  - Fri May  3 15:02:59 2013
Tasks: 190, Power (W): 45.2563, CPU (%):100

  PID   |        COMMAND          |  %CPU   |  %MEM   |  DISK_IO  |  PE_MMC  v|
11362   |stress                   | 40.5797|0.0039543|          0|   13.5721|
11420   |stress                   | 22.2222|  19.2426|     122880|   7.48472|
11421   |stress                   | 21.7391|  16.7917|     155648|   7.32453|
11186   |ectop_case_demo          | 4.34783|0.0903262|   30535680|   1.55754|
2616    |chrome                   | 1.93237|  2.54537|  398880768|  0.756566|
2240    |pulseaudio               | 1.93237| 0.224567|    4612096|  0.756566|
11      |events/0                 |0.966184|        0|          0|  0.436178|
1958    |Xorg                     |0.483092|  1.53263|   19206144|  0.275984|
3453    |java                     |0.483092|  12.3651| 2966568960|  0.275984|
11476   |gnome-screensho          |0.483092| 0.517398|   16097280|  0.275984|
4116    |chrome                   |0.483092|  1.68477|   18792448|  0.275984|
1647    |dbus-daemon              |0.483092|0.0678487|    1470464|  0.275984|
11489   |gnome-screensho          |0.483092| 0.038295|    1323008|  0.275984|
3675    |gnome-terminal           |       0| 0.393564|   26857472|  0.115789|
2486    |multiload-apple          |       0| 0.106352|     913408|  0.115789|
12      |events/1                 |       0|        0|          0|  0.115789|
2484    |cpufreq-applet           |       0| 0.132159|    2519040|  0.115789|
2483    |gnome-keyboard-          |       0| 0.109057|    1351680|  0.115789|
        |                         | 96.6184|  74.2067| 5240005632|   54.0388|
```

Fig. 1. `ectop` command line interface

3.3 Application's Energy Profiling

For the profiling of applications an application's energy profiler, namely Valgreen [10] is available. It uses `libec` to aid the programmer to implement energy efficient algorithms by sampling the power consumption of a given application in small time steps. These time steps may be configured and the smaller it goes, more precise the energy measurement will be.

3.4 Dynamic Power Estimators

Auto-generated models have been widely used on the field of application's power estimation. This use case will compare a dynamic model that is updated through linear regression in regular time intervals with a simple static model. These dynamic models do not need any input from the user and can be used with different devices. The idea is to show the importance of a dynamic model when the workload on the machine changes and our model is not valid anymore.

Figure 2 presents a comparison between a dynamic model, the actual power measured with a wattmeter and a static model parameterized in another machine. One can see that as time goes by, the adaptive model gets closer to the actual dissipated power. Here we show the total power consumption of the machine, but the same model can also retrieve the power dissipated by each process.

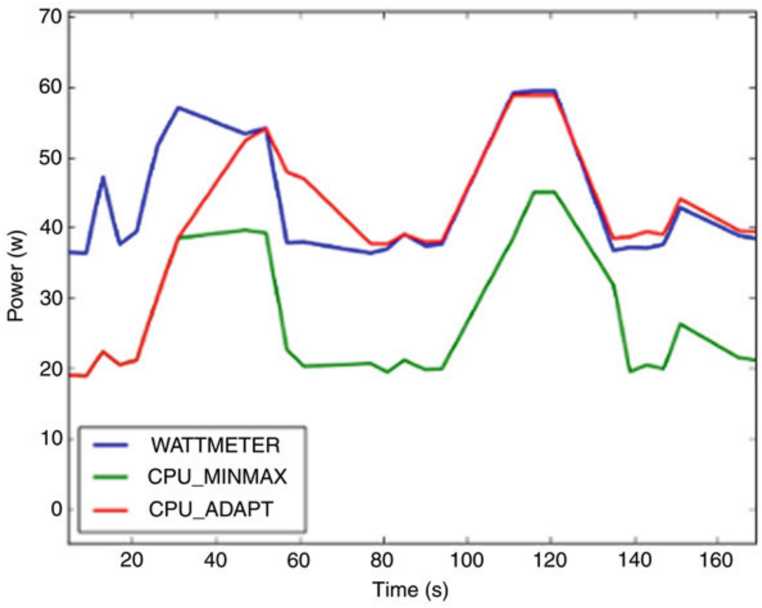


Fig. 2. Comparison between a wattmeter, an adaptive model (CPU_ADAPT) and a static model (CPU_MINMAX).

4 Conclusions

The Energy Consumption Library (`libec`) is a library that can be easily extended and can be used for several different purposes, such as, to compare power estimators in real time, profile applications and efficiently allocate resources taking into account its energy consumption. It is usable on laptops as well as servers and can estimate the power spent by an application even without the presence of a wattmeter.

For the time, `libec` has very few power estimators and it sensors only runs on Linux platforms. As future work we aim to implement new power estimators, as well as expand it to android and windows.

Acknowledgments. The results presented in this paper were funded by the European Commission under contract 288701 through the project CoolEmAll and by the COST (European Cooperation in Science and Technology) framework under Action IC0804.

References

1. Vereecken, W., Van Heddeghem, W., Colle, D., Pickavet, M., Demeester, P.: Overall ICT footprint and green communication technologies. In: 4th International Symposium on Communications, Control and Signal Processing (ISCCSP), pp. 1–6 (March 2010)
2. Rivoire, S., Ranganathan, P., Kozyrakis, C.: A comparison of high-level full-system power models. In: Proceedings of the 2008 Conference on Power Aware Computing and Systems HotPower'08. USENIX Association, Berkele, p. 3 (2008)
3. DaCosta, G., Hlavacs, H.: Methodology of measurement for energy consumption of applications. In: 11th IEEE/ACM International Conference on Grid Computing (GRID), pp. 290–297 (October 2010)
4. Chen, H., Shi, W.: Power measuring and profiling: the state of art. In: Ahmad, I., Ranka, S. (eds.) Handbook of Energy-Aware and Green Computing, pp. 649–674. Chapman and Hall/CRC, Boca Raton (2012)
5. Witkowski, M., Oleksiak, A., Piontek, T., Wglarz, J.: Practical power consumption estimation for real life HPC applications. Future Gener. Comput. Syst. **29**(1), 208–217 (2013)
6. Cupertino, L.F.: Energy consumption tools web-page (www.irit.fr/~Leandro.Fontoura-Cupertino/ectools/) (April 2013)
7. Linux man-pages project: perf_event_open(2) (February 2013)
8. Cappello, F., Caron, E., Dayde, M., Desprez, F., Jegou, Y., Primet, P., Jeannot, E., Lanteri, S., Leduc, J., Melab, N., Mornet, G., Namyst, R., Quetier, B., Richard, O.: Grid'5000: a large scale and highly reconfigurable grid experimental testbed. In: The 6th IEEE/ACM International Workshop on Grid Computing, p. 8. IEEE (2005)
9. Christmann: Description for Resource Efficient Computing System (RECS) (2009)
10. Cupertino, L.F., DaCosta, G., Sayah, A., Pierson, J.M.: Valgreen: an application's energy profiler. In: Conference on Soft Computing and Soft Engineering (SCSE) (2013)

Gicomp and GreenOffice – Monitoring and Management Platforms for IT and Home Appliances

Mateusz Jarus(✉) and Ariel Oleksiak

Poznań Supercomputing and Networking Center, Noskowskiego 10, Poznań, Poland
jarus@man.poznan.pl

Abstract. Due to growth of energy consumption by HPC servers and data centers many research efforts aim at addressing the problem of energy efficiency. Hence, the use of low power processors such as Intel Atom and ARM Cortex have recently gained more interest. In this article, we compare performance and energy efficiency of cutting-edge high-density HPC platform enclosures featuring either very high-performing processors (such as Intel Core i7 or E7) yet having low power-efficiency, or the reverse i.e. energy efficient processors (such as Intel Atom, AMD Fusion or ARM Cortex A9) yet with limited computing capacity. Our objective was to quantify in a very pragmatic way these general purpose CPUs using a set of reference benchmarks and applications run in an HPC environment, the trade-off that could exist between computing and power efficiency.

1 Introduction

Rising energy prices and the increasing environmental awareness of the general public are playing an ever-increasing role in the world of IT and computational science. The push for increased energy-efficiency is also present in the field of desktop computing and even home appliances. All major modern operating systems feature sophisticated power management capabilities based on open industry standards. While these features can reduce the power consumption of a running machine, they do not address the problem of computers which are idling. A typical desktop PC, according to our measurements, can consume between 40 and 90 watts while idling without any gain in productivity.

Moreover, people's awareness on power usage of home appliances is usually very low. Bulbs, TV sets and IT equipment are often left turned on unnecessarily, drawing much energy. A universal interface that presents the power usage of all home devices would be highly desirable to make people aware of how much energy is wasted every day.

We present two platforms for saving energy at home and office. The first one is Gicomp [1,2], an easily extensible, standards-based distributed monitoring and control system for energy consumption management. The main purpose of its creation was to combine various power management related tools, solutions and monitoring equipment using a standardized, easy-to-deploy network protocol.

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 58–62, 2013.
DOI: 10.1007/978-3-642-40517-4_5,

Moreover, we also show Green Office, a project aimed at monitoring and managing the power usage of home devices – computers, lamps, printers etc. Through an easy to use web-based interface the energy consumption of every device at home can be monitored in real-time.

2 Gicomp Architecture

The Gicomp architecture (Fig. 1) integrates various tools and solutions across different levels ranging from integrated measuring devices to OS power management policies. Each of them is represented in the Gicomp platform as a separate Node having its own JID (Jabber ID) in the XMPP network. Every Node plays one of three roles: Device (machine) node, Waker node or Meter node. The role played by the Node is determined by the handlers, software plugins described shortly in Sect. 3, that are enabled. The Device node is responsible for interacting with the machine's operating system. It provides our platform with the power management capabilities already available on the machines. The Waker node is responsible for bringing up machines after their shutdown by the means of Wake-on-LAN, IPMI or other vendor-supplied means. Meter nodes are interfaces between the SMOA Devices platform and the power measurement/control hardware. The type of hardware is irrelevant, it could be some embedded measuring devices, manageable power distribution units (PDU) or intelligent server cases. The node is installed on every managed machine. It hardly uses any resources, does not display any messages and never interrupts the user's work. Its job is to constantly monitor the computer by checking the utilization of the computer's components, observing user activity, collecting statistics concerning energy consumption. Based on this knowledge it manipulates computer settings, such as monitor brightness, power modes of whole computer or some components (e.g. hard drive), etc. Information about its work is gathered by the server and

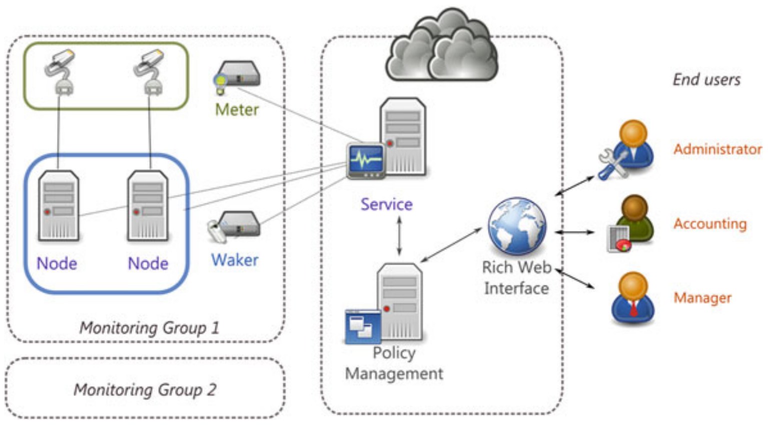


Fig. 1. Gicomp architecture.

may be displayed to the administrator. Moreover, different power plans can be determined to define how the computer reacts to user activity. An easy to use, friendly and nice-looking interface was developed to access all these settings. It is accessible through web-browser and is protected by password. Moreover, SSL encryption is used to transfer data. Gicomp allows to create graphs of energy consumption for every computer or groups of computers. Thanks to the use of Round Robin Database, historical data is available up to one year back for every computer in the company. The power usage is estimated based on current system state – CPU load, monitor brightness and specified minimum and maximum values of a given computer. However, more elaborate techniques will be also available soon, especially for laptop devices. These include an automatic and accurate detection of machine's power draw.

3 Green Office Architecture

Green Office enables the automated control of energy consumption at home or office. All devices and computers are connected to intelligent sockets which are monitored and controlled remotely via ZigBee protocol. In this way, actual power usage of every single device can be observed and also controlled remotely, for example, by adjusting light intensity.

Because all of the data is collected wirelessly, no additional cables are required. Using our software we can check detailed power consumption of every device, change the brightness of lamps and turn on and off computers. When integrated with RFID (Radio-frequency identification), we can detect and identify people entering/leaving offices and perform appropriate actions - such as turn on preferred computer and adjust the brightness of lamps (or just turn everything off when the person leaves the office).

All of the data collected by ZigBee module is forwarded to another computer for further processing. To reduce the power consumption, an ARM device with a maximum power draw of a few Watts can also be used. It also gives access to a web-based user interface that shows the state of every connected device, their current power usage and enables to turn on and off computers and change the brightness of the lamps.

4 Demonstration

In this demo we present Gicomp, Green IT COntrol and Management Platform, and Green Office, a project that helps monitoring and managing power usage of devices at home.

Gicomp demonstration consists of a few computers, connected either remotely or on-site to our server that collects information about their energy consumption and allows the administrator to manage them. The management panel (Fig. 2) is available through web-browser and shows graphs with the historical power usage, allows to control the state of the machines and also create power plans that define how the computer reacts to user activity.

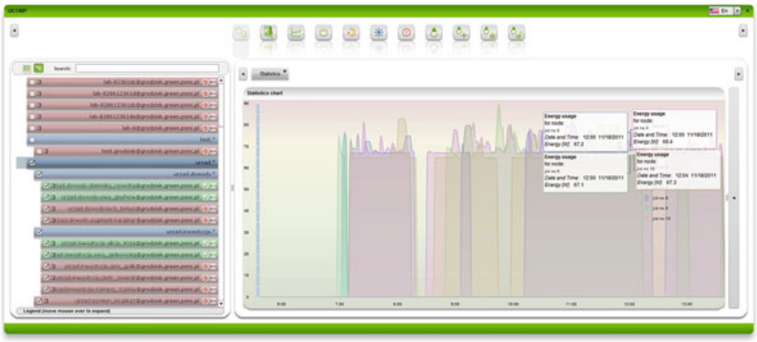


Fig. 2. Gicomp WebGUI.

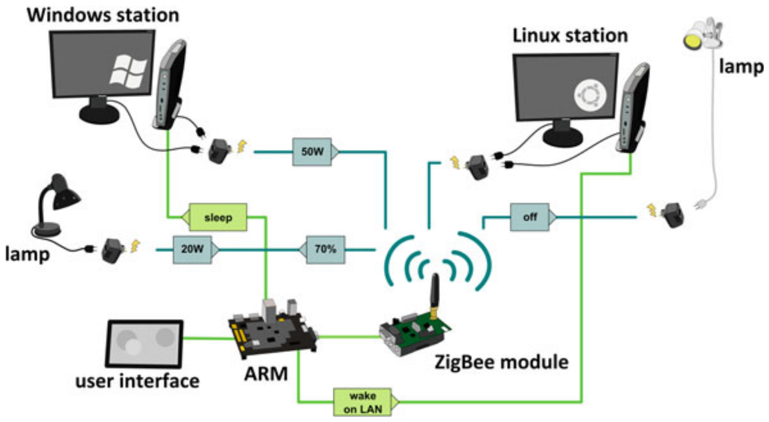


Fig. 3. An example of devices connected to the Green Office platform.

The Green Office demonstration scheme is presented in Fig. 3. It consists of two computers and a few lamps connected to the electricity through our intelligent sockets. They collect information about power consumption and transfer them wirelessly to the ZigBee module. This data is later processed by another low-power computer, that also provides an interface to view the statistics and perform operations on the devices. The interface can be accessed remotely, on another computer. In this case a tablet is used that connects to the internet wirelessly.

5 Conclusion

Gicomp is an ideal system for saving energy in offices with large number of computers. With an easy to use interface, it is possible to monitor and manage machines from a centralized management panel. Its wide-ranging monitoring and control capabilities make Gicomp a well-suited solution for many application scenarios in the upcoming years of increased environmental awareness and the green computing revolution.

Green Office is a similar solution but provides fine-grained monitoring and control of home appliances. With the use of intelligent sockets, it enables energy consumption measurements and device steering, such as switching them on/off, adapting light intensity and more. It also allows the user to monitor the power draw of devices in real-time and collect historical statistics.

These solutions can be combined to control every device at home or office, measure their energy consumption and change their state. Instead of software power estimation currently used in Gicomp, Green Office intelligent sockets can be used to retrieve accurate power consumption of computers. Such a combined platform would be very versatile, giving control over all appliances.

References

1. Kurowski, K., Oleksiak, A., Witkowski, M., Nabrzyski, J.: Distributed power management and control system for sustainable computing environments. In: Proceedings of the International Conference on Green Computing GREENCOMP '10, pp. 365–372 (2010)
2. Minkowski, A., Oleksiak, A., Witkowski, M.: Monitorowanie i zarządzanie zużyciem energii przez komputery oraz urządzenia w infrastrukturze administracji samorządowej przy użyciu systemu GICOMP. In: Wybrane Aspekty Rozwoju Lokalnego Kapitalu Ludzkiego w dobie Spoleczeństwa Informacyjnego (2011)

Modelling Power Adaption Flexibility of Data Centres for Demand-Response Management

Andreas Berl(✉), Gergö Lovász, Ferdinand von Tüllenburg, and Hermann de Meer

Computer Networks and Communications, University of Passau, Passau, Germany
{andreas.berl, gergoe.lovasz, ferdinand.tuellenburg, hermann.demeer}@uni-passau.de

Abstract. Demand-response management is an approach that includes the power demand side into the power management process to reshape power demand of consumers to the current availability of power. Data centres are major energy consumers that are highly interesting for demand-response management. However, in contrast to many other energy consumers, data centres have a highly dynamic flexibility in terms of power adaption, depending on the current situation, which makes their integration into demand-response management difficult. This paper suggests a model for the dynamic power adaption flexibility of data centres, to foster their integration into demand-response management.

1 Introduction

The current power distribution grid was originally not designed to handle growing energy demand [1], reduce CO_2 emissions, be energy efficient, or integrate decentralised power generation based on highly volatile renewable energy sources. In spite of these challenges, power supply needs to match power demand as closely as possible, to keep power quality (e.g., in terms of current, frequency, and voltage) on a high level. *Demand-response (DR)* management establishes a communication flow between energy provider and energy consumer to enable the reshaping of power demand. *Open Automated Demand Response (OpenADR)* [2], e.g., provides a widely known DR solution.

Data centres provide great opportunities in being integrated into DR management, due to their high power demand.[1] Also, according to a study that has been performed for the Californian Energy Commission [3], data centres have a significant potential for DR that is not yet exploited. Various *power adaption strategies* [3,4] to exploit the flexibility in power demand are available within the data centre: the virtualisation and consolidation of services, shifting of services in time, migration of services across data centres, management of hardware energy-saving features, storage of energy in Uninterruptible Power Supplies (UPS), or the management of air-conditioning. It is important to see, however, that the

[1] http://www.guardian.co.uk/sustainable-business/data-centres-energy-efficient

DOI: 10.1007/978-3-642-40517-4_6,

power adaption flexibility depends on the current situation within the data centre. The degree of consolidation depends on the number of running services and the current usage of these services, for instance. The flexibility achieved by the cooling system depends of the heat within the data centre and probably outside weather conditions, and the shifting of jobs in time depends on the current mix of jobs in the data centre. Most of the resources that are used for the described strategies need to recover after a power adaption, e.g., the UPS batteries need to be recharged as fast as possible to have full reaction capacities to possible blackouts, or, in the cooling approach, the temperature needs to be brought back to normal operation temperature to prevent hardware damage. In other cases a recovery is not necessary, e.g., if services have been consolidated or hardware features have been used to fulfil a request of the energy provider. Instead, the services are de-consolidated again and the hardware performance is turned back to normal. In these cases, power adaption is achieved by reducing the Quality-of-Service (QoS) provided to data centre customers. This lost QoS can not be recovered afterwards and has to be covered by special service level agreements with the customers. In contrast to other approaches (e.g., the OpenADR temporal model for DR Events [5]), this paper suggests a flexibility model that is is able to cope with dynamic power adaption flexibility.

2 Power Adaption Flexibility Model

A DR approach that includes data centres needs to consider their dynamically changing power adaption flexibilities. This means that the energy provider should not request a predetermined power adaption with a specified height and duration, but cope with the currently available flexibility within the data centre. To achieve this, the data centre's current flexibility needs to be communicated to the energy provider in a formal way. The flexibility of the data centre can be expressed by modelling available strategies in terms of the power adaption they are currently able to achieve. Each of these models expresses the current power adaption flexibility of the data centre for a selected strategy. The data centre can send a set of models to the energy provider, who is able to chose the most suitable model from this set for DR management.

Figure 1 illustrates an example of a power adaption strategy as dashed line. The x-axis shows the time in minutes and illustrates different adaption phases, the y-axis shows the adaption of power consumption in kW. It can be observed that the power adaption during normal operation is around 0 – no adaption is performed. During the active phase, the power consumption is significantly reduced. During the recovery phase, the power consumption is higher than during normal operation of the data centre (e.g., due to the recharging of UPS batteries).

In the suggested model, the active and recovery phase are approximated by a step function $f(t)$. This has the advantage that the model can be easily determined and described by only using a few parameters. Although this simplification does not reflect all fluctuations in the power adaption of the data centre, it is assumed that the accuracy is sufficient to achieve DR management. This assumption is based on the fact that there are also fluctuations during the normal

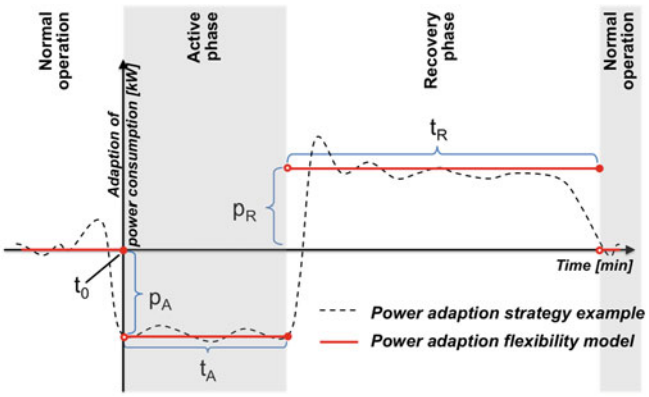


Fig. 1. Power adaption flexibility model

operation of the data centre that can not be controlled by the energy provider. These kind of fluctuations are put into perspective by fluctuations caused by all other energy consumers that are connected to the grid. Six main parameters determine the characteristics of the step function $f(t)$:

- t_0 represents the starting time of the adaption specified in the adaption request which is sent by the energy provider.
- p_A is the decreased/increased power consumption during the active phase (compared to normal operation), measured in kW. p_A is positive if the power consumption is required to be increased, and p_A is negative if the power consumption is required to be decreased.
- t_A is the length of the active phase in minutes.
- t_{gap} is the length of the gap between active and recovery phase in minutes.
- p_R is the decreased/increased power consumption during the recovery phase, transition phase, and exit phase (compared to normal operation), measured in kW. $p_R = 0$ if no recovery phase is required, p_R is positive if p_A is negative, and p_R is negative if p_A is positive.
- t_R is the overall length of recovery phase, transition phase, and exit phase in minutes. $t_R = 0$ if no recovery phase is required.

With these parameters, the step function $f(t)$ can be defined as:

$$f(t) = \begin{cases} p_A & \text{if } t_0 < t \leq t_0 + t_A \\ p_R & \text{if } t_0 + t_A + t_{gap} < t \leq t_0 + t_A + t_{gap} + t_R \\ 0 & \text{otherwise} \end{cases}$$

The power adaption flexibility model is also illustrated in Fig. 1 (step function). In this example it is assumed that $t_{gap} = 0$. During the active phase, the data centre reduces its consumption on average by p_A kW, while during the transition phase, recovery phase, and exit phase it has on average an increased power consumption of p_R kW. The step function based model describes the power adaption flexibility of a power adaption strategy with only 6 parameters. The model enables data centres to easily communicate the flexibility that is achieved by different power adaption strategies to the energy provider. The energy provider has to chose an appropriate set of power adaption strategies and to aggregate strategies of different data centres to solve the DR management.

3 Discussion

The suggested flexibility model for data centres is able to provide an approximation of the flexibility that is achieved by different power adaption strategies within the data centre. This model can be used to compute the current flexibility within the data centre as well as to communicate the flexibility to the energy provider. The energy provider is able to select and aggregate a suitable subset of strategies across different data centres. On request, a data centre offers one or more appropriate power adaption strategies to the energy provider. A possibility to further regulate such offers would be to send two strategies with maximised properties: the strategy with the longest duration and the strategy with the highest power adaption. However, both of the strategies would need to fulfil the minimum power reduction that has been agreed on by contract between energy provider and data centre. Additionally, the suggested model allows for the negotiation of strategies with the energy provider. In a case where the data centre is not able to fulfil a concrete request of the energy provider due to its current situation, it may still be able to offer an alternative strategy, lower or shorter than the original request.

Acknowledgments. This research has been supported by the EU FP7 Project All4 Green "Active collaboration in data centre ecosystem to reduce energy consumption and GHG emissions", FP7-ICT-2011-6.2.

References

1. Battaglini, A., Lilliestam, J., Bals, C., Haas, A.: The supersmart grid. In: European Climate Forum, Potsdam Institute for Climate Impact Research (2008)
2. Piette, M.A., Ghatikar, G., Kiliccote, S., Koch, E., Hennage, D., Palensky, P., McParland, C.: Open automated demand response communications specification (version 1.0), California Energy Commission (2009)
3. Mares, K.: Demand response and open automated demand response opportunities for data centers (2010)
4. Klingert, S., Berl, A., Beck, M., Serban, R., di Girolamo, G., de Meer, H., Salden, A.: Sustainable energy management in data centers through collaboration. In: Husko, J., de Meer, H., Klingert, S., Somov, A. (eds.) E2DC 2012. LNCS, vol. 7396, pp. 13–24. Springer, Heidelberg (2012)
5. Piette, M., Ghatikar, G., Kiliccote, S., Koch, E., Hennage, D., Palensky, P., McParland, C.: Open automated demand response communications specification (version 1.0). Technical report, Ernest Orlando Lawrence Berkeley National Laboratory, Berkeley (2009)

Part II

Distributed, Mobile and Cloud Computing

StressCloud
A Virtualized Infrastructure Load Injection Tool

Guillaume Le Louët(✉) and Jean-Marc Menaud

Mines Nantes, Inria, Lina, 4, rue Alfred Kastler, BP 20722,
44307 Nantes Cedex 3, France
glelou10@mines-nantes.fr

Abstract. The interest in *Cloud Computing* has grown steadily over recent years, leading to a more intensive use of the datacenters, an increase in their power consumption and a growing complexity of their administration tasks. To assist the administrator in his tasks, the managers of the virtualized datacenters now implement parametrized administration policies, like the Dynamic Power Management in the VMW are software suite. The evaluation of these policies becomes an important and difficult issue for their developers and the datacenters administrator, as the power and performance models of the datacenter's environment become increasingly complex with the new virtualization techniques. Those techniques become so complex that there is a need for load injection frameworks able to inject resource loads in a tested datacenter instead of model-driven simulators. In this article we present *Stress-Cloud*, a framework to manipulate the activities of a group of Virtual Machine managed by a Virtualized Data Center Manager and observe the resulting performances. This extensible framework allows to describe, via a scripting language, a scenario of system resources use in Virtual Machines, execute it on a Infrastructure as a Service Provider containing modified Virtual Machines and retrieve the performance data of those Virtual Machines. The resources managed are the CPU, storage and network accesses with extensibility in mind. We show that our load injection model is reliable and the language is expressive enough to describe complex scenarios, with the example of one NASGrid benchmark. We also show that this load injection framework can be used to model hosts' power consumption based on the hosts' resources activities.

1 Introduction

During the last years, the increase of the IT investment, operating and maintenance costs led many industrials to outsource a part of the their production IT environment to datacenters, buildings dedicated to the execution of informatics services. As the number of services increases, the administration of those datacenters becomes more complex. The services virtualization facilitates this

This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

DOI: 10.1007/978-3-642-40517-4_7,

administration by reducing the number of required hosts. This technique consists in the execution on a host of a dedicated software named hypervisor and responsible of the execution of several Virtual Machines (VMs), abstract services environments. It permits useful operations on the services (suspend, clone, save of the VMs, monitoring of the services). Those operations are handled by Virtualized Data Center Managers (VDCMs), for high-level VMs management. A VDCM monitors the activities of the datacenter : temperature, power consumption, CPU load, RAM usage, storage I/O and network traffic of the VMs and hosts. It analyses those activities and can deduce actions to execute on the datacenter. Those actions can modify the datacenter state, such as the creation and execution of the VMs, their priorities over the computing resources, the power state of the hosts. They may be deduced by the datacenter in order to optimize specific metrics, such as power consumption, while ensuring the validation of the administrator's constraints, or directly requested by the administrator. The number of VDCMs and their functionalities have kept growing with the size and the technological advances of the datacenters, especially since the advent of the *Cloud Computing.*

Several approaches are used to validate or compare VDCMs' policies, from the simulation of the datacenter after a modeling phase to the load injection on real services. If simulations are able to produce results on a large number of datacenters, the model of some metrics can be so complex, such as power consumption, temperature or request response time, that the simulation of those models can become unpractical. What's more, the datacenter administrator will rather validate an algorithm using load injection on a real infrastructure than use generic models that may or may not correspond to its datacenter. To perform a load injection is however not an easy task : the administrator must define the sequence of loads to inject, modify the VMs in which to inject this load and be able to specify each resource activity independently. This paper discusses the execution of a distributed a load scenario in a virtualized datacenter, either to compare several datacenters or to assist in the configuration of their management policies. In cloud terms, this paper considers the datacenters in the Infrastructure as a Service (IaaS) context, meaning we consider a datacenter as a service to execute clients' VMs. As such we do not have any sight nor control of the physical environment and the hypervisors.

This paper makes the following contributions:

– we present the details of a distributed system-level load injection framework
– we present an implementation of such a framework developed in `Java` and `Groovy`
– we evaluate the ability of our framework to specify load injection scenarios.

In this paper we consider the CPU, the network and the storage resources but our framework can easily be extended to control other resources' activities. Our tests show the stressers we developed ensure the injection of a reliable load with small CPU overload in the manipulated VM. Finally, we show our framework can be used to model the power consumption of a host based on this host's resources activities, as proposed in the literature.

Section 2 shows a state of the art on the evaluation of a VDCM's policy and the load injection in a datacenter. Section 3 describes the functionalities and the architecture of *StressCloud*, then in Sect. 3.1 the stressers implementations and in Sect. 3.2 the manipulation language. An evaluation of *StressCloud* is proposed in Sect. 4 and we conclude in Sect. 5.

2 Related Work

Evaluate an administration policy of a VDCM is a difficult task. In previous work [1] on the dynamic consolidation of VMs for energy minimization, we evaluate the administration policy mainly via simulation of a datacenter. We used a simulator to test and validate the algorithm on a large number of configuration, but it did not consider some complex side effects, such as network overload due to multiple VMs migrations. Evaluate an administration policy on real infrastructure with VMs consuming real system resources is closer to the final execution environment and allows us to identify and analyze potential side effects.

For this it is necessary to inject a load on the infrastructure and verify the behavior of the VDCM's administration policy. We consider two load injection levels on a host, depending on the knowledge of its applications: the low-level system resources load injection and the high-level applications load injection. The low-level load injection aims at controlling the use of the host's physical resources, such as CPU, hard drive or network card and allows to check the stability of the system under a given resource load. Those resources access are performed using stress applications, which are dedicated to the access of one resource. The high-level load injection aims at reproducing real applications loads, for example by simulating on a web server the HTTP requests from multiple users, to ensure e.g. the response time or the fault rate of the application under given request loads. Therefore, the high-level load injectors are specific to the application they reproduce (High Performance Computing (HPC), N-tiers Web, banking simulation, data mining) and the low-level load injectors specific to the resources they control (CPU, storage, network, RAM, GPU).

The low-level injection is used to evaluate VDCM policies on very specific use cases. For example, the system VGreen presented in [2] by Gaurav et al. retrieves the VMs' activities informations in a datacenter and deduces a reorganization of the VMs on the hosts. It is evaluated on a test center having 3 hosts and several VMs, each VM executing a pre-defined static benchmark using either CPU or CPU and memory accesses and powered on when there is a need for increase resource access. Similarly, the Enacloud [3] system aims at allocation and balancing loads in a virtualized datacenter. It is tested on a 60 hypervisors center, each of them executing VMs with HPC benchmarks, compilation applications or webservers, started and stopped at predefined times. However, those system load injectors are not meant to be dynamically controlled in distributed environments and as such can not produce datacenter-wide load injection, like high-level injectors do.

Typically, the Mistral [4] system analyses system resources and power consumption of VMs and optimizes the power consumption in the datacenter. The authors evaluate their algorithm on a datacenter running the RUBIs benchmark, on which an external host sends requests to the multi-tier architecture executed locally. This benchmark simulates the activities of multi-tier VMs, simulating response to clients queries. The work presented in [5] presents a center reconfiguration system using a genetic algorithm. This system is tested with a center of 300 hypervisors and VMs running a web host, to which a centralized transmitter sends requests. This evaluation is close to our goal, however the need for the transmitter does not allow scalability and what's more the CPU and network activities of each VM are strongly coupled. The approach in our view the most interesting remains the load injector CLIF presented in [6] and JStress [7], which uses load injectors to modify the activity of the VMs manipulated. However, this approach requires specific load injectors for each load profile to inject and the knowledge of the applications run in a VM.

All these works propose an evaluation of their solutions, but to our knowledge there is no system yet working at IaaS level and allowing both to finely control the system resources consumption, describe and execute complex parametrized load injection scripts on a distributed infrastructure, observe the effective injected load performances and replay these scripts on the same infrastructure. This observation led us to propose a new concept of evaluation of the IaaS and an implementation, named *StressCloud*, presented below.

3 StressCloud : A Framework of Distributed Activities

Our goal is to control and synchronize system-level resources activities of a group of VMs, in order to describe and execute a distributed infrastructure load injection scenario. In this context, a load scenario is a script that describes the resource activities to apply to VMs. Our contributions include the development of resources stressers, the centralized control system and the load scenario language. This framework is made of three parts working together: The *stressers* are run in a VM and are responsible of the access on one of this VMs's system resource. They connect to the *registar*, a controller mechanism for registration, selection and reservation of VMs. This registar is used by the *script executor* which handles the incoming user scripts, sending the stressers' their activities requests and monitoring their performance data.

Our system does not address the VMs provisioning (creation), we assume each VM required for the load scenario is deployed by the datacenter's administrator. To facilitate this deployment *StressCloud* offers a VM image containing basic stressers and communication mechanisms. While the framework does not consider the datacenter's manager, the registar provides a mechanism for identifying and selecting VMs. This mechanism considers the VMs' virtual environment criteria, such as its memory capacity, the number of CPU cores, or its IP address. Those virtual environment informations ca be used by the administrator to identify the manipulated VMs to the center's VMs. Our framework is based on the self-registration of the VMs on a centralized server: each VM connects to the registar, specifies its virtual configuration and stressed resources and waits for user's commands, as presented in Fig. 1.

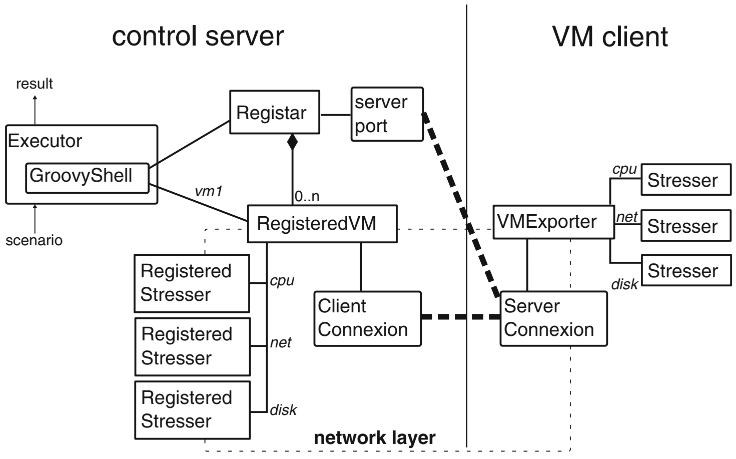


Fig. 1. *StressCloud* architecture

Once the VMs are registered, the user specifies its scenario to select and remotely control the stressers embedded in those VMs. A usage script is a list of instructions specifying VMs' resources activities and sequencing between those activities. We consider the resources activities requests can be either a total amount of operations to perform on the resource or an amount of operations to perform each second. A stress sequence interconnects VMs activities modifications using time constraints or precedence of activity. The script executor registers the stress modifications and their resulting performance values to allow statistic performance analysis for a scenario. Before describing the script language (see Sect. 3.2), the following section presents the operating principle of the stressers proposed by *StressCloud*.

3.1 Generation of System Loads

In the literature, a resource stress is defined by the rate, as a percentage of max capacity, of the resource use. However in a IaaS system the VDCM can migrate running VMs in an heterogeneous infrastructure so the notion of load in the form of percentage does not make sense. Indeed, we want to simulate the behaviour of an application, which produces a given number of accesses per second and a varying resource accesses rate, as expressed in percentage of the host's capacity, when the VM is migrated from one host to another. In this paper, we define a resource activity as a number of accesses to perform on this resource and we distinguish two cases. The first case needs to perform a *work* as quickly as possible, the second case needs to access the resource at a *constant* rate over a given period. In the first case the stresser use the resource at the maximum rate until the amount of work required is complete. A work simulates the behavior of a scientific computing application, we call this type of activity an *absolute load*. In the second case the stresser (e.g. network) is responsible to periodically run a number of operations (e.g. send 10Mb) with a given frequency (e.g. every 10 ms) during a defined period (e.g. 10s). This type of activity can simulate, for example, the behavior of a n-tier application. We call this type of activity a *continuous load*.

At any time, the activity of a VM's stressed resource is either zero, a defined activity in accesses per second or a defined number of work remaining to achieve. We consider three resources currently used by VM: the CPU, disk and network.

All the stressers in *StressCloud* have been developed in Java and are based on the same principle: each stresser periodically execute a number of operations determined by the load demand of the user. The period of their cycle is configurable. In the case of a continuous load, the stresser executes the stress load fraction (a number of operations proportional to the length of the cycle) at the beginning of the cycle, then stops after execution of this fraction until the next cycle. In the case of an absolute load the stresser executes a fixed number of operations per cycle but immediately resumes the next cycle without stopping until the required amount of executions is complete.

To compare the behavior of VDCMs and verify that the required load has been executed correctly, we added the stressers the ablity to return either a error rate for continous load or the number of remaining executions for absolute load. The error rate is the number of times per second the stresser could not run the required number of operations in a cycle time, corresponding to the saturation of the access to the resource. This way we can measure the reactivity of the VDCM and evaluate allocation policies.

One of the difficulties in the development of the stressers was the cycle duration: a cycle too long causes significant threshold effects misleading the VDCM, while a cycle too short can result in high errors and abnormal side effects, e.g. typically a high-frequency disk access has a significant impact on the CPU. Another issue was the meaning of the COU activity to inject, as different processor architecture could lead to different observed processor instructions per second. The following section describes how the scripting language can express the stressers activities.

3.2 Control of the Load with a Script Language

In order to describe scenarios, we developed a scripting language for selecting the VMs to handle, then specify their activity sequences and finally retrieve the scenario's execution performances. This language is an extension to the `Groovy` language [8], reducing the burden of creating our own language syntax and execution environment. We use the GroovyShell `Java` class as it stands for the Groovy runtime environment and can receive and execute Groovy scripts as well as binary code calls. This environment implements the management of variables, method invocation and access to internal objects. Our language thus benefits from all these features.

We distinguish three phases in the development of a scenario. The first phase is responsible to identify and select the VMs required for the execution of the scenario. The second phase is responsible for specifying the loads, whether continuous or absolute, within VMs and the synchronization between those loads. The last phase consists in the retrieval of the performance data resulting from the scenario's execution on the selected VMs.

VMs selection The VMs provisioning phase is not supported by *StressCloud*, as this part is very specific to each IaaS system. We consider the administrator installs *StressCloud* in existing VMs or deploys specific Vms image executing *StressCloud* on the tested IaaS, modify the VMs according to the scenario's needs (number of vcpu, memory size etc.), then start all the VMs needed for the script to execute. The VMs started with *StressCloud* register themselves on a deposit (called registar) centralizing all the informations of the VMs instantiated (vcpu, RAM, IP address etc.). The first phase of the script describes the selection of the VMs needed to run the scenario, using the characteristics of the VMs.

The `group = require (criterion number)` method is responsible for this operation. It reserves a `number` of VMs with respect to the characteristics `criterion`. The criterion can be `null`, in this case no specificity is required on the VMs and any unreserved VM may be selected, which is useful when working on an homogeneous datacenter. The criterion may relate to the number of COU cores allocated, the size of the RAM or the IP address of the VM. The method returns an array (`set`) of VMs. To avoid any interruption in the script, a *timeout* is configurable. If at the end of the *timeout* the system failed to select the number of VMs necessary, the method returns `null`.

After using the VMs for the execution of the script, the `release (group)` releases the VMs reserved in `group` and sets their activities to 0. Calling this method without setting free all VMs the registar reserved.

GroovyShell proposes useful features for collection manipulation, allowing easier manipulation of the VMs obtained by `require`. e.g. `g1 = group [0 .. 9]; g2 = group [10 .. 99]` defines g1 as the first ten elements of `group` and g2 as the 90 following.

Manipulation of the VMs' resources' activities Once the VMs are selected, some functions allow the user to manipulate the system activities in the VMs. As our approach is generic, the `vm.getTypes()` lists the system resources being manipulated within the VM (i.e. all stressers available). The stressers then give access to two manipulation functions: the setting of a *continuous load* or the addition of an *absolute load*, named work.

The assignment operator in our language `load = vm.res` sets the use of the resource `res` of the VM to a *continuous load* `load`/s. To differentiate the two types of loads, we used another operator for the assignment of the *absolute load* : `vm.res+work` adds a quantity `work` of accesses to perform by the resource `res` of VM `vm`. The execution of this work begins immediately if there is not already running work. These mechanisms are generic for all the resources to stress.

The network stresser operates on the same principle, however, we needed to express the emission of data from one VM to another one. Indeed, linking VMs by the network is an important feature of the language and deserves simplifications in order to reduce the size of the scripts and thus the work of the developer. Thus, `vm.send (vm2, size)` requires sending the VM `vm` to send, via the stresser `net` a quantity `size` data to the IP address of the VM `vm2`.

To enable the script description, it is necessary to describe the sequences between the activities, based on time constraints or on precedence. This feature is described below.

Sequencing the activities After review and evaluation, we determined that all of the scenarios we wish to describe can be limited to two synchronization methods. The first synchronization method starts an activity after some time T, the second at the end of a previously started activity.

Queuing temporal activities is expressed using the `vm.after(seconds)`, making the VM `vm` wait for `seconds` seconds before resuming the execution of its commands. An alternate way to control continous loads is `vm.till(seconds)` which makes the VM wait until the script has started its execution for `seconds` seconds. The starting point of a script is always the last time it used `require`.

The queuing of activities can also be done on a scenario-level: using `after (seconds)`, the scenario execution itself is delayed for given number of seconds are elapsed; however the VMs keep sending their commands. This allows to chain several dissociated scenarios on time.

To ease the description of simple scenarios, we provide the method `lb=load Base(time, mult)` to create a shared load specification `lb`, which will add load activities to a stresser using for example `lb.to(stresser, load1, load2)`. This example will first set the load of the stresser `stresser` to $mult \times load1$ for `time` seconds, then to $mult \times load2$ for the same time, then set back the load to 0.

Work-level synchronization is performed by means of the `vm.after(group)` command, delaying activity modifications of the VM `vm` until all stressers and VMs in `group` have completed all their work requested before the call to `after`.

A shortcut of this method is `vm.stresser.after()`, which makes the VM `vm` wait till its stresser `stresser` finishes its requested work. This allows easy synchronization between a VM's several resources' stressers.

As for temporal queuing, the sequential queuing can be done in a scenario level, as `after(group)` blocks the script execution until all stressers and VMs of `group` have finished their work in progress. This creates synchronization points in the script, for example at the end of the execution of the scenario.

Once we have described our activities and scheduled them, we also want to have results of the scenario available in *StressCloud*.

Monitoring and debugging As we request complex scripts, we need to check the effective scenario played by *StressCloud* and monitor the performances experienced by the manipulated stressers during this scenario. To do this we added scenario statistics in *StressCloud* describing the start and stop time of the requested *work* and *load* and for the formers their execution speed, for the later their error rates during execution.

The observation of the scenario execution starts each time the user reserves VM using `require` and stops at the first `release`. Meanwhile the script executor registers all stressers' *load* and *work* instructions, as well as their effective execution time and the observed error rates of the continuous loads. The performance values are registered every time the stresser's load, work or performances are retrieved and can be retrieved periodically by calling the `vm.ping(seconds)` method.

After the start of the scenario, the `works=works(workers)` method returns all the *work*s registered by the parameter `workers` containing VMs and/or their stressers. This method returns a list of `Work` objects, each of them specifying their start time, end time and requested work value. The list `works` can then be printed in CSV format using `works.toCSV()`, e.g. `works(vm1.cpu, vm2).toCSV()` will return all the works of the `cpu` stresser of the VM `vm1` and of all the stressers of the VM `vm2`.

The method corresponding to the statistics collection for continuous loads is `loads(stressers)`. It produces a list of the loads requested on each stresser of `stressers`. Besides the start time, end time and requested load value, the Load Object contains error rate informations, which are stored every time the VM exits from `after(seconds)` method or the `skipped` value of a stresser is requested. The `toCSV()` method prints the requested loads, associated to the error rates.

If the scenario execution behaves badly, besides the `release()` method, several methods allow to monitor and stop a VM's activities. The method `vm.res.skipped` returns the actual error time of the stresser `res` of VM `vm`. This error time is the number of activities *ms* skipped due to the load being higher than the resource's capacity provided to the VM. For absolute work, the `vm.res.work` returns the remaining number of actions to perform before finishing the stresser `vm.res`'s work. Once the VM with incorrect behavior is identified, `vm.cancel()` cancels pending orders on VM `vm`. This prevents from waiting too long for a work followed by a `after()`. The call `vm.clear()` cancels pending orders on VM `vm` and sets its stressers' activity to 0. Those methods allow to monitor and alter a running scenario, collecting performances and stopping bad behaviors. They are also used to evaluate the correct execution of the stressers on a datacenter.

4 Evaluation

We evaluate this framework from two angles, from the stressers points of view and from the language executor point of view. We first check that our basic stressers implementations actually use the system resources (CPU, network and disk) in relation to the requested load. We also check the performance monitors record correct performance values. Using those performance values we then use this framework to produce consumption data from a host and deducing the power used by several parts of a host. Then we describe a NASGrid scenario and a WebServer scenario in our language, to ensure its descriptive capacity. The NASGrid scenario is used on a small datacenter after a benchmark phase to ensure the VDCM can monitor the requested loads the stressers received.

4.1 Resources Stressers

The aim of a stresser is to periodically access one the resources of its VM. We thus need to verify the relation between the load assigned to a resource stresser and the effective use of this resource. As some resource access may produce activity on another resource (typically CPU), we also need to check the CPU usage induced by the stresser's activity. We evaluate the CPU, network and disk stressers confidence as well as their CPU overload.

CPU stress This evaluation starts a CPU stresser and observe the VM's CPU load with system tools, such as “top” on a linux server or the “Management-MXBean” Object in Java. We vary the CPU load and observe both the CPU time used and the error rate detected by the stresser. Each modification of an internal values is followed by a 10s wait to allow the system to reach stability, then we record the error rate for 2s. The server's Dynamic Voltage and Frequency Scaling (DVFS) is disabled as it may modify the meaning of the observed CPU usage depending on the DVFS state of this CPU. This evaluation was performed using 4 stresser threads and a 100ms granularity. The result is shown with the disk and net CPU impact in Fig. 2 and shows that the CPU load of the process is linear with the requested load, up to a certain threshold (here 8000), from which the CPU load growth is reduced while the error rate detected begins to grow. The maximum load seems to be 300% of the capacity of one core, while using 4 cores we expected 400%. This singularity may be due to the allocation of multiple threads on a single core. This test validates the ability of our implementation to perform a constant CPU load for low load goals.

Network and disk stress The evaluation of the disk and network stressers was similar to CPU stresser's. We executed the DISK and NET stressers and changed the requested load, observing the system's effective resources usage. The range of loads was fixed using a first bench, we used a percentage of the known max load for each stresser. To verify the impact of a resource stresser on the VMs's CPU activity, the CPU activity is monitored at the same time as the system disk and network loads.

The result of this evaluation, as shown in Fig. 3, presents an effective load linear with the one requested, till this requested load reaches the capacity of the stresser (here, between 40000 and 60000 kB/s) or network. For the disk stresser we notice a CPU load varying from 5 to 7 %, for the network stresser we notice a CPU load lower than 3% of the CPU. the high CPU activity induced by the DISK stresser may be due to the handling of the partition system in the OS, meaning each disk access results in partition management. These evaluations

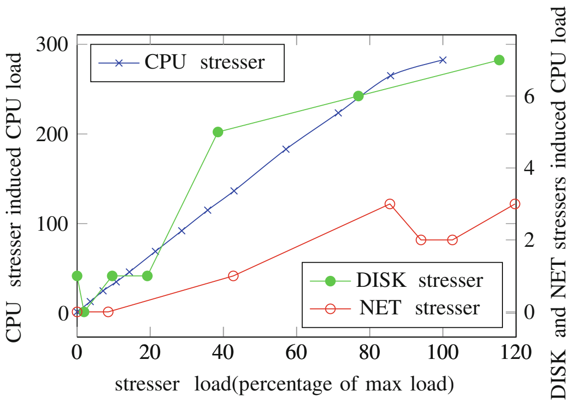


Fig. 2. CPU usage induced by the stressers. DISK and NET stressers use the right axis.

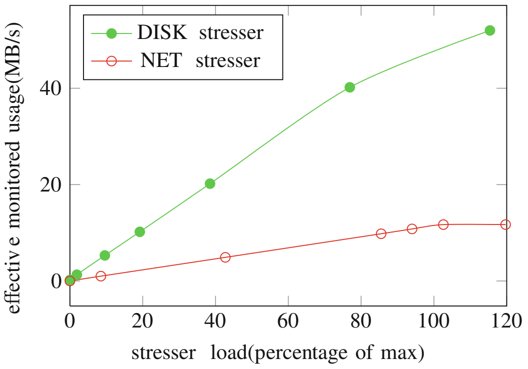


Fig. 3. Effective resource usage induced by the DISK and NET stressers.

confirm that our implementation of resource stressers is suitable for our use, despite CPU overload is not negligible. We also evaluated the ability of our implementation to detect errors in the application of a requested load.

Error monitoring In this evaluation we inject CPU load in a computer using the framework with one VM registered. During the scenario we change the CPU load, retrieve the error value after a 100s adaptation period and compute the maximum load of the CPU stresser using this error value, i.e. $max = request \times (1 - error)$. The host is a HP-Z200 desktop computer, the VM running a ubuntu OS The script used is presented along with the results in Fig. 4. This result shows the error rate retrieved in *StressCloud* is constant through varying loads once those loads reach the resource load threshold and the error rate is null before that threshold. According to those values, the use of the monitored error rate is

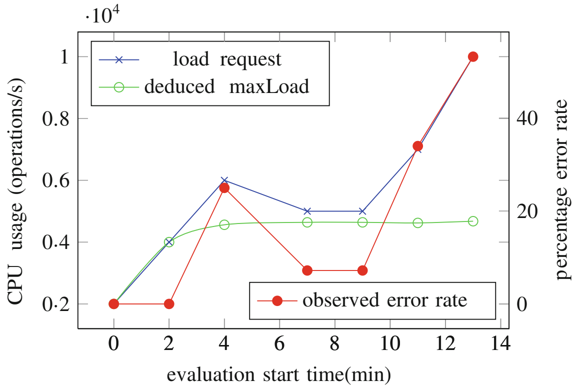


Fig. 4. CPU load request, error rate(right axis) and max CPU evaluation.

a valid indicator of the scenario execution and of the resource capacity provided to the VM under stress.

script used :

```
release();base=2000;period=2
vm1=require(null)
vm1.ping(100)
lb=loadBase(period, base)
lb.to(vm1.cpu, 1,2,3,2.5,2.5,3.5,5)
after(period*7+1);loads().toCSV()
```

Once we showed that the stressers create requested loads in a host and they can observe the error rate induced by their load, we need to verify we can observe power consumption variations induced by load request variation in a host.

4.2 Power Modeling

We used this scripting tool to observe the power consumption variations of hosts under varying resources load. We executed a increasing resource activity scenario on several VMs hosted on one poweredge M100e and monitored the CPU activity and power consumption of the host. The result, presented in Fig. 5, shows the evolution of the host power consumption with the monitored resources requested activities. It shows that on this model of host, there is a relation between the CPU activity induced by *StressCloud* and the observed power consumption of the host. As such, our framework can be used to guess the parameters of a power consumption model by describing a varying resources activities scenario.

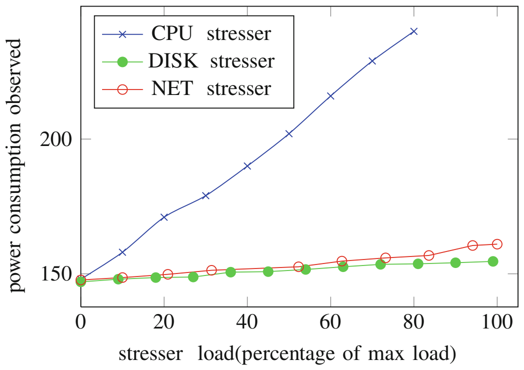


Fig. 5. Power consumption monitored with requested stressers load.

4.3 Scenario Description

Once we know our framework can produce a correct activity on one VM's resource, we need to evaluate its capacity to describe activity scenarios containing several VMs.

Webserver scenario Our first scenario consists in the load increase of a 3-tier web service application. In this application, one VM receives incomming requests, delegates them to tier-2 VMs and two 3rd-tier VMs are responsible of the storage of the Tier-2 VMs. The overall load of the aplication increases for an hour, then decreases for half an hour.
The values of base CPU load in the tiers are deduced from previous observations.

```
release();t2nb=20;t3nb=2;
period=60*6;
cpuT1=100;cpuT2=300;netT2=10000;cpuT3=100;diskT3=netT2*t2nb/t3nb
loads=[0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1,0.9,0.8,0.7,0.6,0.5]

t1=require(null);t2=require(null, t2nb);t3=require(null, t3nb)
lbT1cpu=loadBase(period, cpuT1)
lbT2cpu=loadBase(period, cpuT2)
lbT2net=loadBase(period, netT2)
t2.eachWithIndexit, i->it.net.target=t3[i%t3nb].ip
lbT3cpu=loadBase(period, cpuT3)
lbT3disk=loadBase(period, diskT3)

lbT1cpu.to(t1.cpu,loads)
t2.eachlbT2cpu.to(it.cpu,loads);lbT2net.to(it.net, loads)
t3.eachlbT3cpu.to(it.cpu,loads);lbT3disk.to(it.disk, loads)
```

This complex script shows that *StressCloud* gives the ability to describe complex CPU, network and disk activities at the same time on different VMs.

NASGrid scenario We chose to implement the simplest scenario of the NAS-grid benchmarks suite, the ED scenario[1]. In this scenario, one master node(tier 1) starts running operations, then sends execution data to several nodes in a second tier. One node in a third tier waits for each node of the second layer to finish its execution before doing a final computation. This scenario can be described with the following script:

```
amount=500000; tier2size=2
tier1=require(null);tier2= require(null,tier2size);tier3=require(null)

tier1.cpu+amount
tier2.each{it.after(tier1).cpu+3*amount}
tier3.after(tier2).cpu+amount

after(tier3);works().toCSV()
```

This script is very simple but ensures us we can describe correctly the scheduling of HPC activities. The use of a script language also allow users to create templated scripts, modifying e.g. the `amount` variable to increase the duration of the stress. As we used a variable to specify the number of VMs in the `tier2`, we can easily run the stress on a larger number of VMs by setting this value to a higher number.

This script produces two results: the first is the list of requested works with their start and finish times. This table is created by *StressCloud* when we require the list of works after a scenario:

resource	start	end	work
vm4.cpu	761	944	500000.0
vm3.cpu	183	761	1500000.0
vm2.cpu	183	761	1500000.0
vm1.cpu	0	183	500000.0

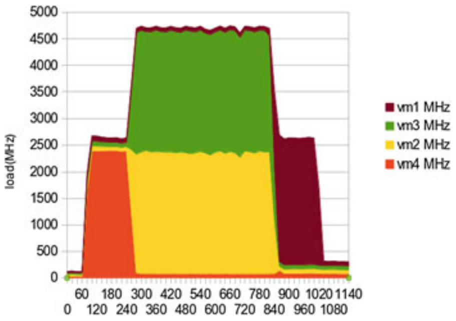


Fig. 6. server load(MHz) on time (s)

The second result, presented in Fig. 6 is the graphical activity of the VMs observed by a VDCM. We used the VDCM VMWare's Vcenter5.0. The VMs were running Ubuntu server VMs.

Those results show that the requested sequence of activities was correctly observed in the VCenter monitor.

[1] See http://www.nas.nasa.gov/publications/npb.html

5 Conclusion

The constant growth of *Cloud Computing* use for production and testing environment has led to the development of a large number of Virtual Machines Managers (VMMs), more and more complex. The evaluation of these systems, responsible for managing the entire environment and reducing the datacenter power consumption is a crucial point for administrators. However, the specific and independant evaluation of these VDCMs on a real center actually received little attention in the literature. Indeed, it requires to have a tool to script the load increase of a center.

In this paper we propose *StressCloud*, a solution of load injection at Iaas script to finely control system resource loads in a group of VMs. Unlike related work, our framework allows to manipulate individual access to the VMs' resources CPU, network and disk via a central repository and retrieve the scenario's execution's performance in order to compare several executions.

We have shown that our language can describe not only load increase scenarios in a center of WebServer type, but also complex load sequences scenarios a center of type HPC. This tool can perform reproductible load manipulation scenarios to e.g. evaluate the impact of a VDCM's algorithm on a datacenter or model the power consumption of the hosts installed in the datacenter.

References

1. Hermenier, F., Lorca, X., Menaud, J.M., Muller, G., Lawall, J.: Entropy: a consolidation manager for clusters. In: Proceedings of the 2009 ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, VEE '09, pp. 41–50. ACM, New York (2009)
2. Dhiman, G., Marchetti, G., Rosing, T.: vgreen: a system for energy-efficient management of virtual machines. ACM Trans. Des. Autom. Electron. Syst. **16**(1), 6:1–6:27 (2010)
3. Li, B., Li, J., Huai, J., Wo, T., Li, Q., Zhong, L.: Enacloud: an energy-saving application live placement approach for cloud computing environments. In: Proceedings of the 2009 IEEE International Conference on Cloud Computing. CLOUD '09, pp. 17–24. IEEE Computer Society, Washington, DC (2009)
4. Jung, G., Hiltunen, M.A., Joshi, K.R., Schlichting, R.D., Pu, C.: Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures. In: Proceedings of the 2010 IEEE 30th International Conference on Distributed Computing Systems, ICDCS '10, pp. 62–73. IEEE Computer Society, Washington, DC (2010)
5. Mi, H., Wang, H., Yin, G., Zhou, Y., Shi, D., Yuan, L.: Online self-reconfiguration with performance guarantee for energy-efficient large-scale cloud computing data centers. In: Proceedings of the 2010 IEEE International Conference on Services Computing, SCC '10, pp. 514–521. IEEE Computer Society, Washington, DC (2010)
6. Dillenseger, B.: CLIF, a framework based on fractal for flexible, distributed load testing. Ann. Telecommun. - Annales des TéLécommunications. **64**(1–2), 101–120 (2008)

7. Wensel, Chris K.: JStress: a performance profiling harness. http://jstress.manamplified.org/ (September 2007)
8. Barclay, K.A., Savage, W.J.: Groovy programming. Morgan Kaufmann Publishers, San Francisco (2006)

An Intelligent and Adaptive Threshold-Based Schema for Energy and Performance Efficient Dynamic VM Consolidation

Seyed Saeid Masoumzadeh[✉] and Helmut Hlavacs

Research Group Entertainment Computing, University of Vienna, Vienna, Austria
Masoumzadeh@gmail.com,
Helmut.hlavacs@univie.ac.at

Abstract. Dynamic VM consolidation as a dynamic control procedure is an effective way to improve energy efficiency in cloud data center. In general, the problem of dynamic VM consolidation can be split into four sub-problems (1) host overloading detection; (2) host under-loading detection; (3) VM selection and (4) VM placement. The goal of this procedure is to operate in a way that optimizes energy-performance tradeoff inside cloud data center. To this end, each of aforementioned sub-problems must operate in an optimized manner. In this paper with concentration on host overloading detection sub-problem, we propose an intelligent and adaptive threshold-based algorithm for detecting overloaded hosts by Dynamic Fuzzy Q- learning (DFQL). Our algorithm with interaction of dynamic characteristics of each physical host environment learns when a host must be considered as an overloaded host towards energy-performance tradeoff optimization inside data center. We validated our approach with the CloudSim toolkit using real world Planetlab workload. We show that our approach outperforms the state of the art algorithms.

Keywords: Green cloud · Dynamic VM consolidation · Fuzzy Q-learning

1 Introduction

The cloud computing is a new computing model allows customers the provisioning resources on demand in a pay-as-you-go manner over the Internet, therefore, service providers without any up front investment in infrastructure could simply rent resources from infrastructure providers to its own need and pay for usages. Beside huge economical impact, cloud technology nowadays exhibits a high potential to be a corner stone of e new generation of sustainable and energy efficient ICT. However, virtualization technology in cloud data center has a significant role in decreasing energy consumption by using fewer physical servers with much higher per-server utilization but it introduces some new management challenges since large pool of virtual machine must be provisioned and managed.

In fact, the meaning of energy efficient resource management in a cloud-computing environment is to assign dynamically physical resources to virtual machines in a way that minimizes energy consumption in a data center while keeping quality of service based on SLA. Therefore, researchers in this filed have been trying to design an effective and optimal management system that can be able to satisfy these constraints. Dynamic VM consolidation as a dynamic control procedure is an effective management system to improve energy efficiency in a cloud data center [1]. In general, the problem of dynamic VM consolidation can be split into four sub-problems (1) determining when a host is considered as being overloaded (host overloading detection), in this situation live migration is required to migrate one or more VMs from the overloaded host; (2) determining when a host is considered as being under-loaded (host under-loading detection), in this situation the host is ready for switching to sleep mode, thus, all VMs have to migrate from it; (3) determining which VMs must be selected to migrate form overloaded host (VM selection) and (4) determining which hosts must be selected to place migrated VMs (VM placement). The goal of this procedure is to operate in a way that optimizes energy-performance tradeoff inside cloud data center. To this end, each of aforementioned sub-problems must operate in an optimized manner.

In this paper we concentrate on host overloading detection sub-problem of dynamic VM consolidation procedure and propose the Dynamic Fuzzy Q-learning (DFQL) algorithm [2, 3] in order to tackle this problem. With respect of dynamic nature of cloud environment that is raised by dynamic workload, host overloading detection can be considered as a dynamic decision making task. In a threshold-based approach, a host overloading detection algorithm must decide about numerical value of threshold in an online manner (if the CPU utilization exceeds the threshold, the host is considered as an overloaded host). This decision must be made in a way that optimizes energy – performance tradeoff in the future. Q-learning as a model free Reinforcement Learning (RL) is an effective approach to design dynamic system managements and consequently, produce dynamic decision-making tasks. In principle, Q-learning can automatically learn high-quality decision making tasks without an explicit model, and with little or no built-in system specific knowledge. In this paper our proposed model by interaction with dynamic characteristics of physical host environment learns when a host must be considered as an overloaded host using the threshold based approach, towards energy-performance tradeoff optimization. The superiority of our algorithm in proportion to previous ones is that our model benefits learning procedure that allows it learns an optimal policy from experiments gained by interaction with dynamic characteristics of the physical host environment. The rest of the paper is organized as follows. In Sect. 2 we discuss about related works. Section 3 gives an introduction of DFQL. Section 4 we propose the idea of DFQL model for host overloading detection and finally in Sects. 5 and 6 we present the experimental setup and simulation results.

2 Related Work

In this section we discuss about priory approaches about host overloading detection policy in dynamic VM consolidation proposed in the literatures. In general, there exist two approaches in literatures in order to energy efficient resource management. In the first approach, researchers modeled VM consolidation as a bin-packing problem (a combinational optimization problem) and formulated it with energy and performance constraints. Then, an optimization algorithm (mathematical or meta-heuristic) as a part of a centralized controller is employed to solve it periodically. In this strategy, in consideration of provided solution by optimization algorithm, live migration as a control act, maps Virtual Machines (VMs) to Physical Machines (PMs) in each period in order to consolidate them into the minimal numbers of physical host inside data center. In this approach there is not any overloaded detection strategy. One of the good examples of this approach is [4]. The authors of this paper modeled workload consolidation as a bin-packing problem and proposed the genetic algorithm to solve it. Actually, they presented a periodically dynamic consolidation based on genetic algorithm that is combined with a workload forecasting (by historical data). The objective of proposed approach is to consume the least possible power and fulfill service level constraints. Their central manager is responsible to run genetic algorithm periodically and locates VMs to the optimal place with live migration and finally switch off the idle servers.

In the second approach, the authors mostly have been trying to present a dynamic control procedure into a two levels controller. In this structure the first level, manages resources in the data center level, and the second level usually manages resources in the host level or VM level. The first level and second level operate in a way that optimizes energy-performance trade-off. In this approach, some researchers presented the host overloaded detection algorithm inside the second level management. Some of them are static and some others are dynamic.

Gmatch et al. [5] applied a two levels resource management. The first level controller is a VM placement one and the second level is a migration one. The first level as a trace-based workload placement controller, collects data on resource usage by VMs instantiated in the data center and uses this historical information to optimize the allocation, while meeting the specified quality of service requirements. This controller has two components encompassing an emulator to predict the long-term impact of integrated management policies for realistic workloads and an optimization search component based on genetic algorithm. The second level controller is a fuzzy logic based feedback controller that operates as a reactive migration controller. An advisor module of the controller continually monitors the servers resource utilization and triggers a fuzzy logic based controller whenever resource utilization values are too low or too high. When the advisor detects an underutilized or over utilized situation the fuzzy controller module identifies appropriate actions to remedy the situation. The most prolific researchers in this approach are Beloglazov et al. In [6] they proposed four heuristic algorithms to determine when and which VMs have to be migrated from overloaded host, they apply a static utilization threshold to detect overloaded host, if the CPU utilization exceeds the threshold, it means that host is overloaded. In [1] they introduced several dynamic host-overloading detection algorithms based on historical data from the resource usage by VMs. In this paper, the authors proposed two robust statistic methods namely Median Absolut Deviation (MAD) and Interquartile Range (IQR) in order to estimate the numerical value of the CPU utilization threshold in an online manner. The algorithm decides that the host is considered overloaded if the CPU utilization exceeds the estimated threshold. In addition of aforementioned threshold based approaches, the authors presented a new method based on maximum time required for a migration. In the new method, the authors utilize a local regression algorithm based on Loess method to fit a trend polynomial to the last k observations of the CPU utilization. This trend line used to estimate a next observation. The algorithm decides that the host is considered overloaded if the next observation satisfies an inequality. Finally in [7] they showed for improving quality of VM consolidation, the time interval between VM migrations from an overloaded host must be maximized. To this end, they employed the Markov chain model under an explicitly specified QoS goal and proved their proposed method has superiority in proportion to heuristic-based methods and the methods rely on statistical analysis of historical data.

In fact, our algorithm is categorized into the threshold-based algorithms with this different that our algorithm benefits from experiment gained by learning procedure to decide better about the numerical value of CPU utilization threshold in the future.

3 Dynamic Fuzzy Q-Learning

In a standard reinforcement learning, at each time-step t, the agent as a learner by interaction with its environment observes the current state, s_t and selects an action, a_t from the set of possible actions corresponding to the state. One time-step later, in parts of a consequence action, agent receives a numerical reward, r_{t+1} and finds itself in a new state, s_{t+1}. In a trial and error interaction, finally agent learns how to map states to action, so as to maximize the discounted sum of rewards obtained, which is given by $\sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$, here γ is a discount factor. This mapping denoted by π and is called agent control policy. The problem can be modeled by Markov Decision Process (MDP) and solved by dynamic programming (DP). DP provides algorithms to find optimal policy π^* with corresponding optimal state value function V^* for perfect model of MDP. Optimal policy is the policy with the highest value function for all states of the environment. Q-learning [8] is one of the most popular RL methods. It involves in finding state-action qualities rather than just state value. Assume that $Q^*(s,a)$ be the expected discounted reward of taking action a in state s, and then choosing actions optimally afterward. An estimation of optimal state-action value function, $Q^*(s,a)$, denoted by $Q^{\sim}(s,a)$ and the temporal-difference (TD) is used to update it:

$$Q^{\sim}_{t+1}(s,a) = Q^{\sim}_t(s,a) + a\left[r_{t+1} + \gamma \max_{a'} Q^{\sim}_t(s',a') - Q^{\sim}_t(s,a)\right], \quad 0<\alpha<1.$$

Here α is a learning rate.

In many real life scenarios like our case study, sensory are continues. In this situation the representation of high dimensional state space is difficult. This problem is known as the curse of dimensionality. FQL is an extension of the Q-learning method into a fuzzy environment. FQL by the use of Fuzzy Inference System (FIS) partition each continues state space variable and create fuzzy rules. Each rule in the rule-base has a set of discrete actions associated to it. Each action in each rule has a quality factor (a weight), which will be adjusted through learning phase. Thus, each rule also has a weight vector considering of its action qualities. Briefly speaking, FIS estimates the Q-value function for current state-action pair.

This Q-value function along with the optimal Q-value of the state, calculated in the same way, will be used to compute TD Error. Later on based on the TD Error and update rule of TD learning the action weights will be updated towards gaining more reinforcement, as the case in Q-learning. The agent then chooses actions based on the quality values (weights) of different actions available in action set of each rule, along with an exploration/exploitation mechanism named double ε-Greedy. You can see the FQL structure in Fig. 1 and find all details about it in [2, 9].

One of the enhancements for the FQL structure is to make adaptive fuzzy membership functions by clustering input data. This enhancement is very effective in case studies that their input variables are dramatically changeable. DFQL [3] is one of the enhancement of FQL that employ Fuzzy clustering so as to make adaptive fuzzy membership functions. In our case study we use The FCM (Fuzzy C-Mean) [10] as a fuzzy clustering algorithm. The FCM gathers input data during each interval and classifies data into the given number of clusters, after that it establishes membership functions by making projection clusters onto the various input variables x_i. Figure 2 depicts FCM structure for estimating Gaussian membership functions.

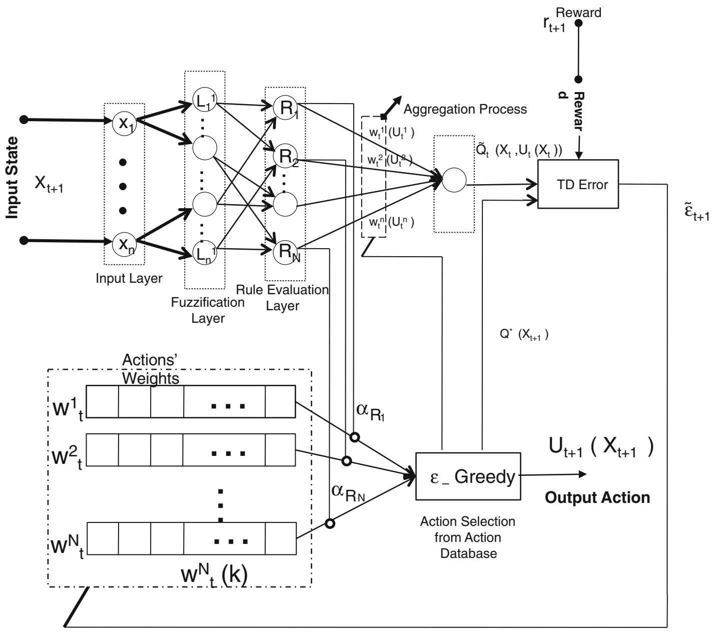


Fig. 1. FQL Structure

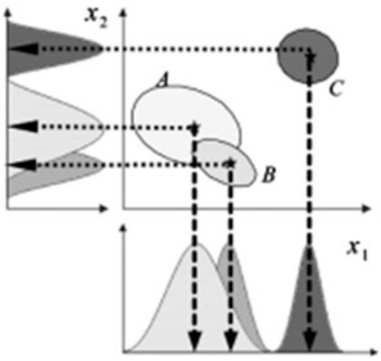


Fig. 2. Adaptive Fuzzy Membership Function by FCM

4 Designing of Proposed Algorithm

Our target systems is an IaaS environment represented by a large scale data center comprising N heterogeneous physical node and is based on CloudSim architecture [11], each node characterized by CPU performance, disk storage, amount of RAM and network bandwidth. The software layer of the system is tiered comprising local and global managers. The local managers reside on each node as a module of VM monitor and involved in monitoring of a node's CPU utilization, resizing the VM according to their resource needs, and deciding when and which VMs have to be migrated from host node. Global manager resides on a master node and gathers information from local managers to adapting allocation of VMs by issuing VM migration commands and changing power state of the node.

In this architecture each of physical hosts decides about its own situation; therefore, it is expected that we embed DFQL as a module into the local managers. Whereas, our problem has a large input space, DFQL in this strategy will suffer from low learning convergence rate. In order to speed up learning convergence rate we embedded DFQL into the global manager as a central module and made a cooperative learning between agents by sharing DFQL's quality weight factor among all physical hosts inside data center. Central DFQL module can employ the local managers monitoring in the global manager for collecting data from host in order to percept current state of each host as well as gathering information so as to calculating reward as a reinforcement signal. In addition it can use the global manager's commanding service to inform local managers about its own decision. The system model has been depicted in Fig. 3.

In Fig. 4 we illustrate a closer view of communication between DFQL and each host. At each time step t, the global manager collects input data from hosts and save them in an input queue. In our implementation, we neglected queuing delay causing execution time of DFQL, therefore, data from each host in the queue is considered as the current state, s_t of the host by DFQL. After processing, DFQL selects a threshold for the corresponding host as an action, a_t. This step is repeated for all hosts in data center. One time-step later, in parts of a consequence action, global manager gathers

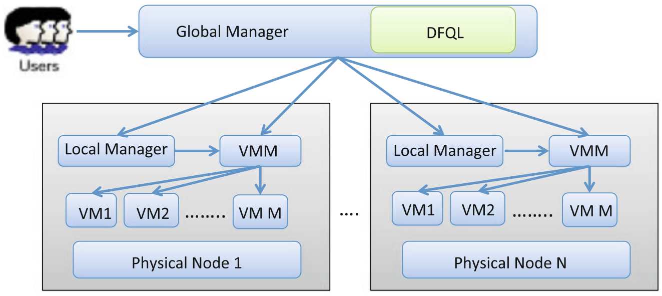


Fig. 3. System Model

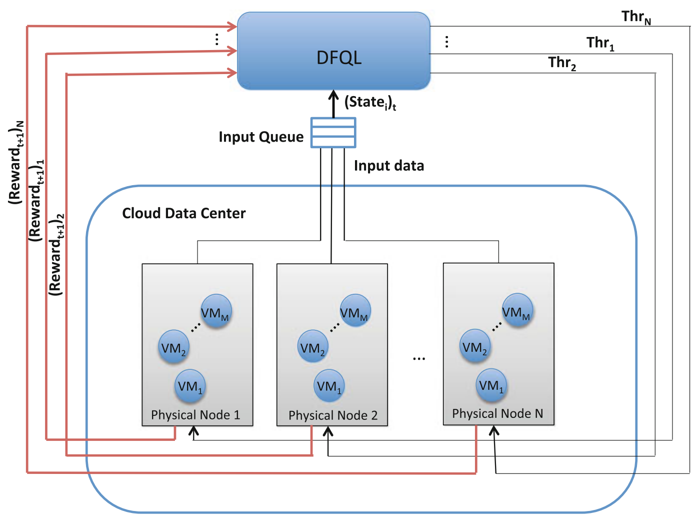


Fig. 4. DFQL Host Overloading Detection

information from each host and the DFQL calculates the numerical reward as a feedback or reinforcement signal. In this architecture DFQL shares its own quality of actions (weights) among hosts. Whereas hosts experience almost the same workload characteristics during their own lifetime, they would be able to contribute each other for updating DFQL weights, which it conclusively led to increase of learning convergence speed.

Formulating Host Overloading Detection in Dynamic Fuzzy Q-Learning. Before formulating host overloading detection as an FQL task we address some metrics and definitions precisely. To present a description of energy-performance tradeoff we must present a definition for energy consumption and cloud performance separately. In our system model the Energy Consumption (EC) by a server is defined as a linear function from CPU utilization, and cloud performance is defined as a function that evaluate the

Service Level Agreement (SLA) delivered to any VM deployed in an IaaS. We apply two metrics for measuring of SLA violation. SLA Violation Time per Active Host (SLATAH), and Performance Degradation due to Migrations (PDM) [1]. These metrics was defined with this assumption that the SLAs are delivered when 100% of the performance requested by applications inside a VM is provided at any time bounded only by the parameters of the VM. In fact SLATAH is the percentage of time, during that active hosts have experienced CPU utilization of 100% and PDM is the overall performance degradation by VMs due to live migration:

$$SLATAH = \frac{1}{N}\sum_{i=1}^{N}\frac{T_{s_i}}{T_{a_i}}$$

$$PDM = \frac{1}{M}\sum\nolimits_{j=1}^{M}\frac{C_{d_j}}{C_{r_j}}$$

Here N is the number of active hosts; T_{s_i} is total time during which host i has experience CPU utilization of 100%; T_{a_i} is total time during which host i being in the serving VMs; M is the number of VMs; C_{d_j} is a estimation of the performance degradation of the VM j caused by migration (10% in our experiments); C_{r_j} is total CPU capacity requested by VM j during its life time. Therefore, a metric for describing SLA violation can be defined as follow:

$$SLAV = SLATAH \times PDM$$

Therefore, the best metric in cloud data center that can be able to describe energy-performance tradeoff is the product of EC, SLATAH and PDM that the authors of [1] denoted it as Energy SLA Violation (ESV):

$$ESV = EC \times SLAV$$

In consideration of aforementioned formulation we can define SLA Time (SLAT) for each host as follow:

$$SLAT_i = \frac{T_{s_i}}{T_{a_i}} \quad 1 \leq i \leq N$$

The goal of the DFQL host overloading detection is energy-performance tradeoff optimization, thus, the FQL task is formulated as following:

The Reward Function. The long-term cumulative reward is the target of DFQL. The reward is a performance feedback on the resulted new selected threshold for each host; therefore, reward can be defined as follow:

$$(reward_{t+1})_i = \frac{1}{SLAT_i \times EC_i} \quad 1 \leq i \leq N$$

The Input State. Average CPU utilization and number of VMs of each host; make the input state set at each time step. To the best of our knowledge, previous approaches for host overloading detection just have employed host CPU utilization for its own decision-making. In our approach, so as to better percept of dynamic host environment we hybridize average host CPU utilization value with number of VMs resided on the host, it causes DFQL has a better control on PDM. The input state is defined as follow:

$$X_t = \{\mathrm{AHCU}_t, \mathrm{NumVM}_t\}$$

AHCU is Average Host CPU Utilization; and NumVM is number of VMs resided on the host.

The Action. Our actions set is discreet and each of its elements denotes the threshold of host CPU utilization, therefore, it can be defined as follow:

$$A(X_t) = \{Thr_1, Thr_2, Thr_3, \ldots, Thr_n\}$$

5 Experimental Setup

In our experiments the CloudSim toolkit has been chosen as a simulation platform as well as we used real life's workload, which provided as a part of the CoMon project, a monitoring infrastructure for PlanetLab [12]. So as to compare our result with the state of the art algorithms presented in [1] we setup our simulation as same as [1]. We simulated a data center comprising 800 heterogeneous physical nodes. Half of which are HP ProLiant ML110 G4, and the other half consist of HP ProLiant ML110 G5 server. The power consumption of selected servers is different at each load level. The frequencies of the servers' CPUs are mapped onto MIPS rating. Each server is modeled to have 1 GB/s network bandwidth and the characteristics of VM types corresponding to Amazon EC2 instance types including High-CPU Medium Instance; Extra Large Instance; Small Instance; Micro Instance. Initialization of VMs is done according to the resource requirements defined by the VM types. However, during the lifetime, VMs utilize less resource according to the workload data, creating opportunities for dynamic consolidation.

We used three different workload data that was collected in three different days. During simulation each VM is randomly assigned a workload trace from one of the VMs from the corresponding day. Table 1 shows the characteristics of each workload data. In dynamic VMs consolidation procedure, for all of experiments we used Minimum Migration Time (MMT) algorithm [1] for VM selection, Power Aware Best Fit Decreasing (PABFD) algorithm [1] for VM placement and for host under-loading detection we used simple strategy that select the host with the minimum utilization compared to the other hosts.

Table 1. Characteristics of Workload Data

Data	Number of VMs	Mean	St.dev	Quartile 1	Median	Quartile 3
Workload1	1052	12.31%	17.09%	2%	6%	15%
Workload2	1516	9.26%	12.78%	2%	5%	12%
Workload3	1078	10.56%	14.14%	2%	6%	14%

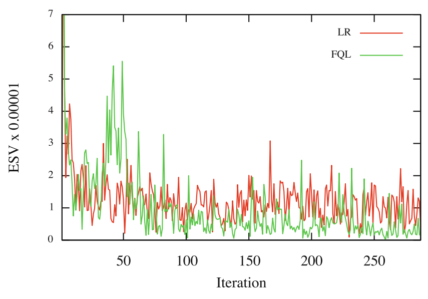
In DFQL setup, we used three fuzzy sets with Gaussian membership function for each of elements of input set. The elements of actions set, was selected experimental as follow:

$$A(X_t) = \{0.5, 0.55, 0.6, 0.65, 0.7, 0.75, 0.8, 0.85, 0.9, 0.95\}$$

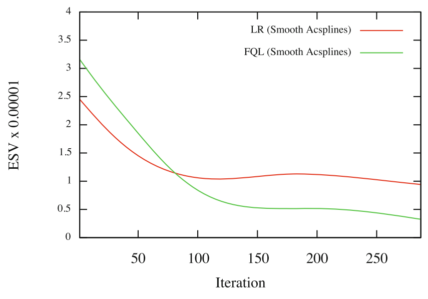
The learning rate of Q-leaning was set to 0.1 and the quality factors of actions (Weights) were initialized randomly. We apply two intervals in our implementation, one of them is 300 second which was used as the FQL iteration or epoch for perceiving next state and receiving reward of previous state, and another one is 3000 second that was used as the FCM iteration for clustering data and changing fuzzy membership function parameters.

6 Simulation Results

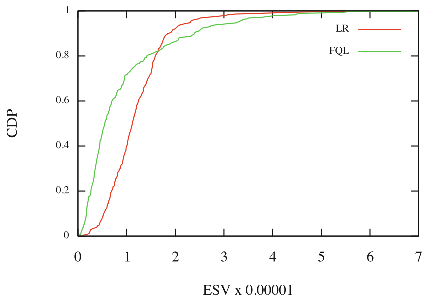
In the first experiment, we compared DFQL algorithm with Local Regression (LR) algorithm, the best among state of the art algorithms that was presented in [1]. Both algorithms make decision about when a host is considered as an overloaded host, but our proposed algorithm has an extra advantage that is to benefit from experiment gained by learning procedure to decide better in the future. In order to illustrate online learning effect in DFQL overloading detection in comparison of LR, we quantified



(a) Discrete ESV Value



(b) Discrete ESV Value (Smooth)

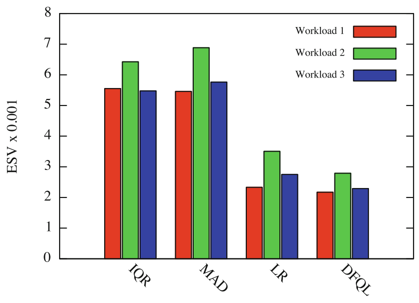


(c) Cumulative Distribution of Discrete ESV Value

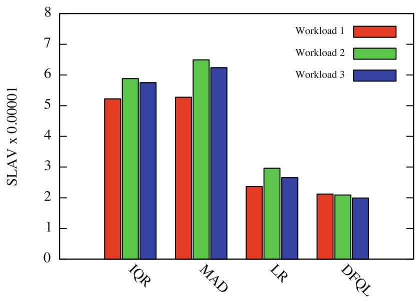
Fig. 5. Discrete ESV Metric

ESV value during every 300 second (DFQL epoch) in one-day simulation with the same workload in Fig. 5. Figure 5a shows the ESV value during each iteration (300 second) in the simulated cloud data center for LR and DFQL. For a better understanding, we made smooth it and illustrated it into the Fig. 5b. The Fig. 5b clearly depicts that DFQL improved its own behavior trough trial and error with interaction of physical hosts' environment and learned to act in a way that brings reward over time.

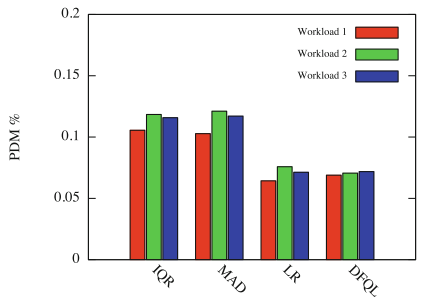
Consequently, it is admissible that the DFQL doesn't have a sufficient performance in the preliminary iterations but after a while it learns a policy to select the CPU utilization thresholds from actions set so as to maximize the cumulative reward.



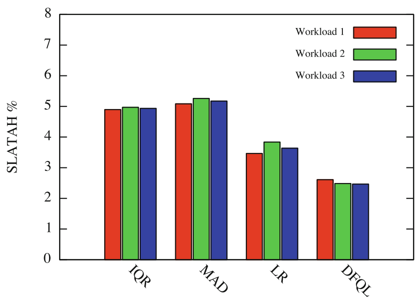
(a) ESV Metric



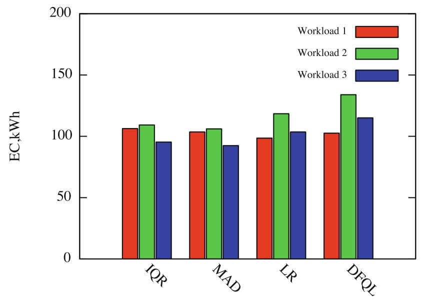
(b) SLAV Metric



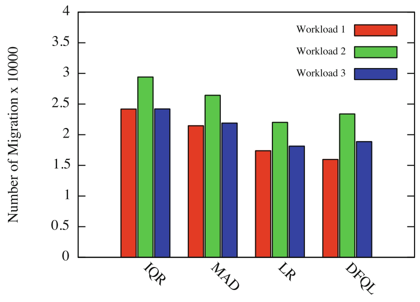
(c) PDM



(d) SLATAH



(e) Energy Consumption



(f) Number of VM Migrations

Fig. 6. Comparison of Overloading Detection Algorithm

Therefore, it is expected that we achieve better and better result in long-term learning. Finally in Fig. 5c we illustrate cumulative distribution of discrete ESV value.

In Fig. 6 we compared our proposed algorithm with some of the sate of the art algorithms that was presented in [1] and we addressed them in the related works section. Figure 6a compared total ESV value amongst four algorithms with three different workload data. It shows that our proposed algorithm in all workloads outperforms other ones. The value of SLA violation (SLAV), SLA Time per Active Host (SLATAH), Performance Degradation due to Migration (PDM), energy consumption and number of migration in Fig. 6b–f respectively. These value as mentioned before, are the constitutive elements of our energy-performance tradeoff model (ESV), therefore, study of them makes clear how our proposed algorithm achieves a better result than other.

The learning procedure in our algorithm will continue until convergence, consequently, it is expected that our algorithm can improve the trade off optimization more in the long-term. Apart from it our algorithm benefits from an exploration/exploitation strategy, therefore, it can change its own policy in the face of major changes in any workload.

7 Conclusion

In this work we presented an Intelligent and adaptive threshold based schema for host overloading detection in dynamic VM consolidation procedure by Dynamic Fuzzy Q-learning (DFQL). Our proposed algorithm by interaction with dynamic characteristics of physical host environment learns when a host must be considered as an overloaded host towards energy-performance tradeoff optimization. The superiority of our algorithm among the state of the art algorithms is that it benefits from experiment gained by learning procedure to decide better in the future. Experimental result shows DFQL outperforms other algorithms and we believe that our proposed algorithm can achieve better and better results during long-term learning. Apart from it, whereas DFQL benefits form an exploration/exploitation strategy it can tackle major changes in workload by changing its policy during cloud data center lifetime. To the best of our knowledge, this is the first work, presents an intelligent schema in host overloading detection in dynamic VM consolidation based on online learning. As a part of our future work we would like to use continues actions instead of discrete actions set, in DFQL structure.

Acknowledgments. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

References

1. Beloglazov, A., Buyya, R.: Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers. Concurr. Comput. Pract. Experience **24**(13), 1–24 (2011)

2. Naeeni, A. F.: Advanced multi-agent fuzzy reinforcement learning. Dalarna University, Dalarna (2004)
3. Juang, C.: Combination of online clustering and Q-value based GA for reinforcement fuzzy system design. IEEE Trans. Fuzzy Syst. **13**(3), 289–302 (2005)
4. Hlavacs, H., Treutner, T.: Genetic algorithms for energy efficient virtualized data centers. In: 8th International Conference on Network and Service Management (CNSM 2012), pp. 422–429 (2012)
5. Gmach, D., Rolia, J., Cherkasova, L., Belrose, G., Turicchi, T., Kemper, A.: An integrated approach to resource pool management: policies, efficiency and quality metrics. In: 2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN), pp. 326–335 (2008)
6. Beloglazov, A., Abawajy, J., Buyya, R.: Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Gener. Comput. Syst. **28**(5), 755–768 (2012)
7. Beloglazov, A., Buyya, R., Member, S.: Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints. IEEE Trans. Parallel Distrib. Syst. **99**, 1–14 (2012)
8. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning : a survey. J. Artif. Intell. Res. **4**, 237–285 (1996)
9. Jouffe, L.: Fuzzy inference system learning by reinforcement methods. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **28**(3), 338–355 (1998)
10. Yang, M.-S.: A survey of fuzzy clustering. Math. Comput. Model **18**(11), 1–16 (1993)
11. Calheiros, R.N., Ranjan, R., Beloglazov, A., De Rose, C.A.F., Buyya, R.: CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Softw. Pract. Experience **41**(1), 23–50 (2011)
12. Park, K., Pai, V.S.: CoMon: a mostly-scalable monitoring system for PlanetLab. ACM SIGOPS Oper. Syst. Rev. **40**(1), 65 (2006)

Energy Characterization of Data Mining Algorithms on Mobile Devices

Carmela Comito[1,2(✉)] and Domenico Talia[1,2]

[1] Center for Informatics and Systems, University of Calabria, Rende (CS), Italy
[2] ICAR-CNR, Rende (CS), Italy
{ccomito, talia}@dimes.unical.it

Abstract. The pervasive availability of increasingly powerful mobile computing devices like PDAs, smartphones and wearable sensors, is widening their use in complex applications such as collaborative analysis, information sharing, and data mining in a mobile context. A key aspect to be addressed to enable effective and reliable data mining over mobile devices is ensuring energy efficiency. In particular, energy characterization plays a critical role in determining the requirements of data-intensive applications that can be efficiently executed over mobile devices (e.g., PDA-based monitoring, event management in sensor networks). Therefore, there is an increasing need to understand the bottlenecks associated with the execution of these applications in modern mobile-based architectures. This paper presents an experimental study of the energy consumption behaviour of representative data mining algorithms running on mobile devices. Specifically, we consider algorithms for association rule mining, clustering, and decision tree induction. Our study reveals that, although data mining algorithms are compute- and memory-intensive, by appropriate tuning of a few parameters associated to data (e.g., data set size, number of attributes, size of produced results) those algorithms can be efficiently executed on mobile devices by saving energy and, thus, prolonging devices lifetime.

1 Introduction

The dissemination and increasing power of wireless devices is opening the way to support analysis and mining of data in a mobile context. Enabling mobile data mining is a significant added value for nomadic users and organizations that need to perform analysis of data generated either from a mobile device (e.g., sensor readings) or from remote sources. Accordingly, an increasing number of cell-phone and PDA-based data intensive applications have been recently developed [1,2]. Examples include cell-phone-based systems for body-health monitoring, vehicle monitoring, and wireless security systems. Monitoring data in small embedded devices for smart appliances and on-board monitoring using nano-scale devices are examples of such applications that we may see in a near future. Support for advanced data analysis and mining is necessary for such applications.

Data mining in such mobile/embedded devices faces various challenges because of several reasons such as (1) low-bandwidth networks, (2) relatively small

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 98–113, 2013.
DOI: 10.1007/978-3-642-40517-4_9,

storage space, (3) limited availability of battery power, (4) slower processors, and (5) small displays to visualize the results. We need to design algorithms and systems that can perform data analysis by optimally utilizing the limited resources.

For mobile devices running on batteries, energy efficiency is a key concern to enable effective and reliable computing over them. In fact, most commercially available mobile computing devices like PDAs and mobile phones have battery power which would last for only a few hours of usage. Therefore, the next generation of mobile applications for such mobile devices should be designed to minimize the energy consumption [3,4].

In particular, energy characterization plays a critical role in determining the requirements of data-intensive applications that can be efficiently executed over mobile devices (e.g., PDA-based monitoring, event management in sensor networks). Therefore, there is an increasing need to understand the bottlenecks associated with the execution of these applications in modern mobile-based architectures. The past few years have seen significant research in reducing the computational complexity of data mining algorithms. Unfortunately, very little has been done to ensure that data mining algorithms are fully usable in a mobile environment. To the best of our knowledge only very few studies have been devoted on energy characterization of data mining algorithms on mobile devices [5].

This paper presents an experimental study of the energy consumption characteristics of representative data mining algorithms running on mobile devices. Specifically, we consider algorithms for association rule mining, clustering, and decision tree induction. The performance analysis depends on many factors encompassing size of data sets, attribute selection, number of instance, number of produced results, accuracy.

To the aim of the performance evaluation, we developed an Android application allowing to execute data mining algorithms over Android smartphones by tuning a set of parameters. To this purpose we establish a benchmarking suite of applications that encompass algorithms commonly used in data mining like the ones above cited.

Our study reveals that, albeit data mining algorithms are compute- and memory-intensive, by appropriate tuning of a few parameters associated to data (e.g., data set size, number of attributes, size of produced results) those algorithms can be efficiently executed on mobile devices so as to save energy and, thus, prolonging devices lifetime.

The remainder of the paper is organized as follows. Section 2 presents the experimental setup used for the energy measurements. Section 3 describes the basics of the data mining category considered for the experimental study. Section 4 presents the performance study. Finally, Sect. 5 concludes the paper.

2 Experimental Setting

This section describes the experimental set-up used for the energy consumption measurements.

As a first step, an Android application has been developed to perform the experimental evaluation. The application has been designed with a twofold objective: (i) integrate the Weka [6] implementation of the reference data mining algorithms in an Android setting; (ii) monitor mobile device resources like battery level, CPU occupancy and memory usage. This last feature has been implemented by exploiting the development tools made available by Android. In particular, the application is able to detect information about resources usage through some files of the Linux kernel of Android. Those files belong to the directory file system *proc* [7] that contains information concerning the current state of Linux kernel, allowing this way applications and users to explore the system status. Within such a directory one can find information about the hardware, like percentage battery level, and each currently running process as CPU occupancy and memory usage.

The energy cost computation is based on battery depletion by exploiting the assumption of proportionality between the percentage of battery level and the energy consumption. Accordingly, the energy cost has been computed as follows. The battery level percentage is gathered through the *proc* files and the corresponding battery level value in mAh is calculated considering the maximum battery level of the device. This value is then converted into electric charge ($Coulomb$) and multiplied to the nominal voltage (equal to $3,7\ V$) to obtain the residual energy of the device, as expressed by the following equation:

$$E(t) = Q(t) * V \tag{1}$$

where $Q(t)$ is, thus, the battery residual charge (Coulomb) and V the nominal voltage.

The above described application once installed on an Android smartphone allows for the (1) execution of data mining algorithms and the (2) gathering of statistics related to the device resources collected during the execution of a given algorithm. In particular, the application through a simple user interface allows for the selection of the algorithm and the data set to be used. Moreover, through the interface it is possible to specify either a set of common performance parameters as data set size or algorithm-specific performance parameters as the number of clusters for the K-means algorithm or the minimum support for the Apriori. The collected statistics are then saved in a SQL Lite database. Note that the input data sets are in the form of ARFF files.

3 Data Mining Algorithms

Data mining is the process of automatically finding implicit, previously unknown, and potentially useful information from large volumes of data [8]. In this work we characterize the energy consumption behavior of three representative data mining algorithms running on mobile devices. Specifically, we consider algorithms for association rule mining, clustering and decision tree induction. Those algorithms are briefly outlined in the following of the section.

3.1 Clustering

Clustering is the process of discovering the groups of similar objects from a data set. K-means is a partition-based method and is arguably the most commonly used clustering technique [9]. Given the user-provided parameter k, the initial k cluster centers are randomly selected from the database. Then, K-means assigns each object to its nearest cluster center based on some similarity function. Once the assignments are completed, new centers are found by the mean of all the objects in each cluster. This process is repeated until two consecutive iterations generate the same cluster assignment.

3.2 Association Rule Mining

The goal of Association Rule Mining (ARM) is to find the set of all subsets of items or attributes that frequently occur in database records. In addition, ARM applications extract rules regarding how a given subset of items influence the presence of another subset.

Apriori [10,11] is arguably the most influential ARM algorithm. Apriori has two phases of execution: candidate generation and support counting. In the first phase all frequently occurring itemsets are identified based on the apriori principle. Candidate itemsets of length k are generated from itemsets of length $k-1$. Frequency of candidate itemsets are calculated and the ones which are infrequent are then pruned to generate frequent itemsets of length k.

3.3 Classification

A classification problem has an input dataset called the training set which consists of example records with a number of attributes. The objective of a classification algorithm is to use this training dataset to build a model such that the model can be used to assign unclassified records into one of the defined classes [8]. Decision trees are a common knowledge representation used for classification. In classification, the decision tree predicts, based on data from a specific instance, the membership class of an instance. Each node in the tree consists of a test, based on one or more attributes of the instance to be classified. The leaf nodes provide the class label.

Among classification algorithms we have chosen J48 that is an open source Java implementation of the C4.5 algorithm in the Weka data mining tool [12].

3.4 Input Data Sets

Input data is an integral part of data mining applications. To support our experimental evaluation, two sample real-word data sets, from the UCI KDD archive [13], have been used as data sources: Covertype with J48; US Census 90 with K-means and Apriori. The first data set contains information about forest cover type for a large number of sites in the U.S. The second one contains data extracted from the U.S. Census Bureau Web site as part of the 1990 U.S. census.

4 Performance Evaluation

The section presents an experimental study aimed at investigating the energy consumption characteristics of representative data mining algorithms. As discussed, we selected three reference data mining algorithms from the Weka library: J48, for data classification; K-means, for data clustering; and Apriori, for association rules discovery. Each algorithm was used to analyze a sample dataset, of varying size, using an Android smartphone as mobile device.

We characterize the performance of those data mining algorithms running over a specific device and with respect to a given data set, in terms of the following metrics: energy consumption, execution time and percentage of CPU used. Each algorithm has been executed 10 times; the values reported in the following of the section are the average values of CPU, energy and execution time consumed.

The results are obtained with two Android smartphones: (1) Sony Xperia P, a 1 GHz Dual Core ARM processor with 1 GB RAM; (2) HTC Hero, a 528 MHz Qualcomm processor with 288 MB RAM. Each test is executed with the following setting: 100% battery level, radio signals off (airplane mode), automatic shut-off display off.

4.1 K-means Performance

In this section we illustrate the results of the study of the energy consumption behavior of the K-means algorithm over the Census dataset. The data set size is 14 MB with 244,348 instances and 11 attributes.

The performance study has been guided by the computational complexity of the algorithm. Therefore, the choice of the performance metrics has been made by looking at the parameters that contribute to the computational complexity of the algorithm. The computational complexity of K-means is $O(nmkt)$ where n is the number of attributes, m the number of instances, k the number of clusters and t the number of iterations. Accordingly, the chosen performance parameters are the: (i) data set size or number of instances, (ii) number of produced clusters; (iii) number of attributes.

In a first experiment, we investigated the impact of the number of instances on the performance of the K-means algorithm. To this aim we scaled down the data set size considering partition of 0.8 MB, 1.6 MB and 3.2 MB, corresponding to 14,446, 28,562 and 56,725 instances, respectively. With higher data set size the mobile device used is not able to execute the algorithm.

Figure 1 shows that the energy consumption scales linearly with the number of instances and the slope of all the curves show a sharp increase for 28,561 instances corresponding to the 1.6 MB data set. The Figure also shows how the number of produced clusters impacts on the performance of the K-means algorithm. We can, thus, evaluate how the contemporary increase of number of instances and number of clusters affects the energy consumption of the devices. In the considered experimental scenario the number of clusters varies from 2 to 32. Figure 1 reveals that, as expected, by increasing the number of instances,

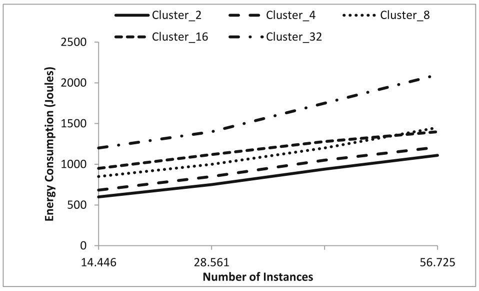


Fig. 1. Energy consumption of K-means w.r.t. the number of instances for different number of produced clusters.

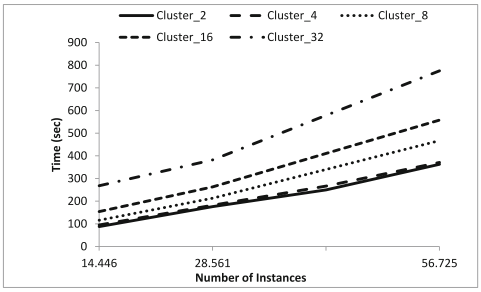


Fig. 2. Execution time of K-means w.r.t. the number of instances for different number of produced clusters.

the energy consumption grows linearly for all the different number of produced clusters. From this result, we can argue that the energy consumption increases faster with the number of instances than the number of clusters. Specifically, the trend is more pronounced from 8 clusters starting from which the grows becomes linear.

Figure 2 shows that the execution time has exactly the same trend as the energy, meaning that the energy consumption is proportional to the execution time. This finding that attests a direct proportionality between power consumption and execution time, also applies to the rest of the experiments for which hereinafter the graphs relating to the execution time will be no longer presented.

The CPU occupancy percentage remains almost constant with the increasing of both the number of clusters and the number of instances, ranging between 91% to 95%. This result confirms that data mining algorithms are CPU bounded (see Fig. 3). For this reason we will not report in this paper more experiments concerning the CPU usage.

We have also evaluated the performance of the K-means algorithm in terms of number of attributes of the data to be mined. For this experiment we considered a data set size of 3.2 MB with 56,725 instances. Figure 4 shows that the energy consumption grows linearly with the number of attributes regardless of

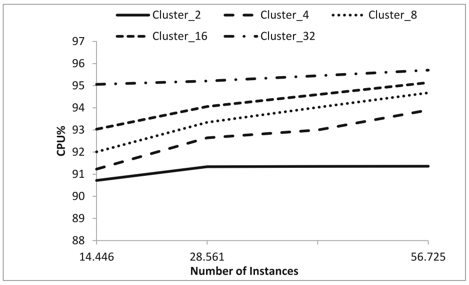


Fig. 3. CPU occupancy percentage of K-means w.r.t. the number of instances for different number of produced clusters.

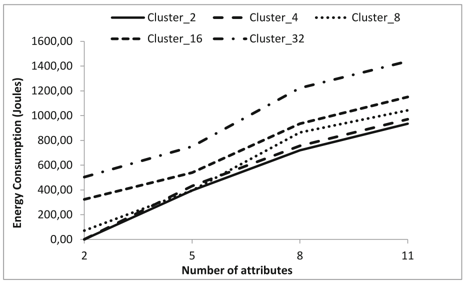


Fig. 4. Energy consumption of K-means w.r.t. the number of attributes for different number of produced clusters for a 3.2 MB data set.

the number of clusters. In particular, for 2 attributes the energy consumption is almost null for 2 and 4 clusters whereas for 16 and 32 clusters is already rather high, equal to about 320 and 500 *joules*, respectively. Starting from 5 attributes the growth becomes substantial also for smaller number of clusters and from this number of attributes the curves associated to 2 and 4 clusters, that up to that point were almost overlapping, begin to detach.

We performed a last experiment to further evaluate the performance of the K-means algorithm. In this experiment we tuned both the number of instances and the number of attributes whereas the number of clusters is set to 32. From Fig. 5, as expected also from previous experiments, is evident that the energy consumption scales linearly with both the number of attributes and the number of instances. Specifically, with just two attributes the energy consumption is almost zero until 14,446 instances, while for higher number of instances it grows sub-linearly. Analogously, in the 5 attributes case, the energy consumption is of few *joules* for 14,446 instances but for higher number of instances it increases linearly. In general, from the experiment is evident that with a number of attributes greater than 2 the energy consumption grows substantially, regardless of the number of instances. The conclusions that can be drawn from this experiment is that with the simultaneous increase of number of attributes and number of instances, specifically from 2 to 8 attributes and for a number of instances

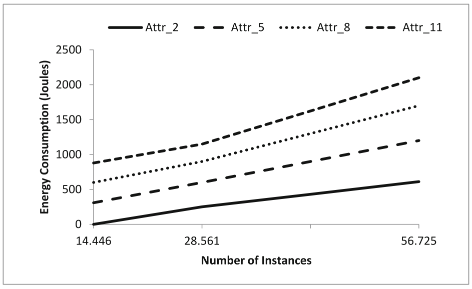


Fig. 5. Energy consumption of K-means w.r.t. the number of instances for different number of attributes.

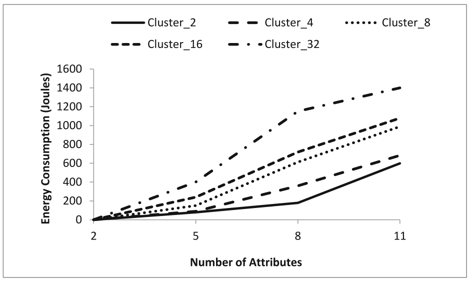


Fig. 6. Energy consumption of K-means w.r.t. the number of attributes for different number of produced clusters for 0.8 MB data set.

greater than 14,446, the energy consumption increases considerably as confirmed by the trend of the graphs that changes from sub-linear to linear.

Combining the result of this experiment with those obtained in the previous ones we can asses whether the simultaneous variation of all the considered parameters (number instances, number of attributes and the number of clusters) causes a variation in the energy consumption that corresponds to the expected one. To this aim we considered three different scenarios in which the energy consumption is evaluated by varying both the number of attributes and the number of clusters. Each scenario is associated to a different data set size. The first scenario is the one with a 3.2 MB data set, discussed above and shown in Fig. 4. The scenarios associated to 0.8 MB and 1.6 MB data sets are briefly illustrated in the following together with the results of the comparison of the outcomes obtained in the three settings.

Figure 6 shows that for a 0.8 MB data set the energy consumption is null for 2 attributes, regardless of the number of produced clusters. Passing from 5 to 8 attributes the energy consumption slightly grows and, then, increases linearly after 8 attributes, when 2 or 4 clusters are produced. On the other hand, starting from a number of clusters equal to 8, the energy consumption grows linearly starting from 5 attributes and showing a considerable increase in the slope in correspondence of 32 clusters. From this result, it appears that for small data

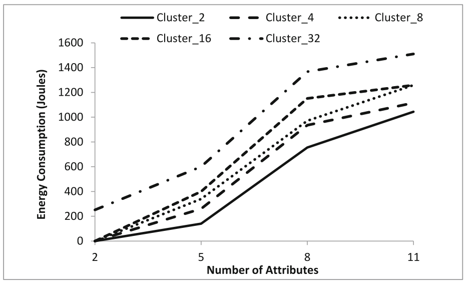


Fig. 7. Energy consumption of K-means w.r.t. the number of attributes for different number of produced clusters for 1.6 MB data set.

set as 0.8 MB, the energy consumption begins to be significant starting from 5 attributes and 32 produced clusters.

In the case of 1.6 MB data set (see Fig. 7), the energy is almost zero when two attributes are considered for all the number of clusters except in the case of 32 clusters where the energy consumption is of about 251 *joules*. Then, starting form 3 attributes, the energy consumption grows linearly with the number of attributes, regardless of the number of produced clusters. Around 5 attributes the growth becomes more pronounced and the graphs associated to the different number of clusters are clearly different. From this experiment it results that for the 1.6 MB data set the energy consumption grows substantially in correspondence of 5 attributes but not for 32 clusters where the consumption is already rather high for 2 attributes.

Comparing the results obtained with the three scenarios, we can notice that increasing the data set size, the energy consumption begins to be felt starting from a number of attributes always smaller. We pass, indeed, from 5 attributes and 32 clusters in the case of 0.8 MB data set to 2 attributes and 8 clusters of the 3.2 MB data set. Therefore, we can conclude that, as expected, the energy consumption increases linearly with the simultaneous growing of number of instances, attributes and clusters showing, however, a smaller increase with the number of clusters.

4.2 Apriori Performance

In this section we present the results of the study of the energy consumption behavior of the Apriori algorithm over the Census dataset. The data set size is 19 MB with 333,011 instances and 11 attributes. Unless otherwise specified, in the experiments we refer to a number of rules equal to 10 with 0.1 minimum support and minimum confidence equal to 0.9.

Before going into the details of the experimental evaluation, we briefly outlines the behavior of the Apriori algorithm to identify the parameters which determine its computational complexity.

The problem of discovering all association rules in Apriori can be decomposed into two subproblems:

1. Frequent itemsets generation i.e. all the itemsets having support greater than the user specified minimum support. The support for an itemset is the number of instances that contain the itemset.
2. Frequent itemsets generated in the step 1 will be used to generate association rules that satisfy user specified minimum confidence. The confidence of an association rule is a percentage value that shows how frequently the rule head occurs among all the groups containing the rule body. The confidence value indicates how reliable this rule is. The higher the value, the more often this set of items is associated together.

If m is the number of instances and n the number of distinct attributes, the number of potentially frequent itemsets is $O(2^n)$ and the overall computational complexity of the algorithm is $O(m2^n)$.

As evident from the algorithm description, the performance characteristic variations of the Apriori are not simply related to the input set characteristics as the number of instances and the number of attributes as indicated in the complexity formula. Rather, other factors like the minimum support and the minimum confidence also have significant influence. For instance, input sets with more attributes tend to generate more candidate patterns, especially in the initial stages of the algorithm where itemsets of relatively small size are being evaluated. However, the number of attributes is not the only factor that impacts on the itemsets generation. In fact, itemsets generated by the algorithm increase drastically at low support values.

Accordingly to the above discussion, the performance parameters chosen are the: (i) number of instances, (ii) number of attributes (iii) minimum support and (iv) minimum confidence.

In a first experiment we evaluated the minimum support considering a 3.2 MB data set with 11 attributes and minimum confidence equal to 0.95. As said before, the minimum support allows for the generation of frequent itemsets taking into consideration as frequent itemsets only the ones having support greater than the user specified minimum support. This phase is the most complex and it is the one that determines the complexity of the algorithm and, thus, the time cost. Figure 8 shows that the energy consumption decreases as the minimum support increases. This is due to the fact that the lower the support, the greater the frequent itemsets generated and the higher the computational cost of the algorithm. Conversely, by increasing the minimum support the number of frequent itemsets generated decreases.

The frequent itemsets generated are then used in the second phase to generate association rules that satisfy user specified minimum confidence. This phase is simpler than the previous one and its complexity is determined by the value of the minimum confidence. To evaluate the impact of the minimum confidence we considered a 3.2 MB data set with 11 attributes and 0.7 as minimum support. Figure 9 shows that the energy consumption slightly decreases as the minimum confidence increases. This is because the larger the confidence, the more often the set of frequent items has to be associated together and, obviously, the higher this value the lower the chance of producing rules meeting the specified confidence.

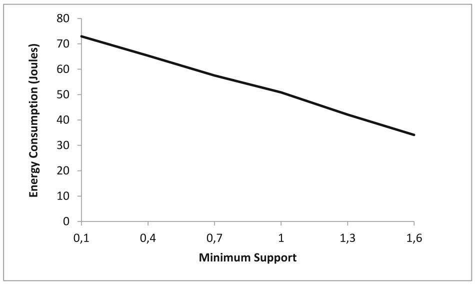


Fig. 8. Energy consumption of Apriori w.r.t the minimum support.

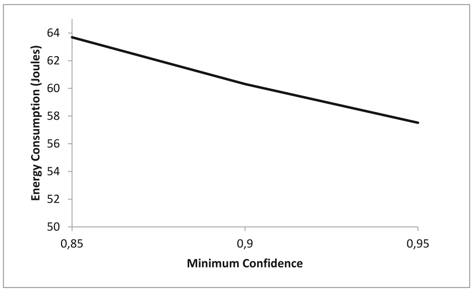


Fig. 9. Energy consumption of Apriori w.r.t the minimum confidence.

In other words, the number of produced rules does not grow with the minimum confidence.

Similarly to what we have done for the k-means algorithm, we evaluated the impact of both the number of attributes and number of instances on the performance of the Apriori algorithm. Accordingly, we scaled both the number of instances and the number of attributes of the discrete version of the Census data set. Precisely, we create the following partitions of the original data set: 0.8 MB, 1.6 MB, 3.2 MB and 6.4 MB corresponding to 14,528, 28,451, 56,682, and 113,377 instances, respectively. This upper bound in the number of instances represents the value beyond which the device is not able to complete the execution of the algorithm.

Figure 10 shows that the energy consumption scales quite linearly with both the number of instances and attributes. One can note that when 11 attributes are used the energy consumption is significantly higher compared to the 5 and 8 attributes cases. On the other hand, till to 28,451 instances the energy consumption is almost the same for the 5 and 8 attributes cases. After this value, the increase in the number of attributes is more pronounced creating, thus, a net margin between the curves associated to the different numbers of attributes. From 5 to 11 attributes, it goes from a gap of a few *joules* for a number of instances equal to 14,528, up to a difference in energy consumption which is almost double in the case of 113,377 instances. For what concerns the energy

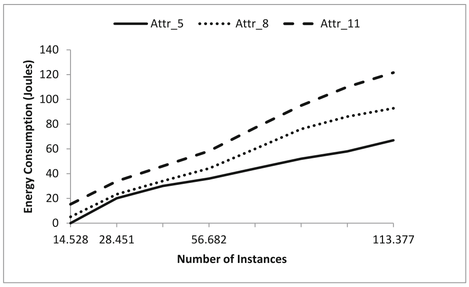


Fig. 10. Energy consumption of Apriori w.r.t the number of instances for different number of attributes.

consumption behaviour when the number of instances grows, it shows a linear trend that remains rather constant with the number of attributes.

The finding that the energy consumption (and the execution time) increases linearly with the attributes may be surprising at a first glance. In fact, from the complexity formula it results that Apriori can be exponential in the number of attributes. However, in this regard two clarifications need to be made. The first one concerns the small size of the used data sets that show very small execution times not suffering excessively by the increase in the number of attributes. The second clarification relates to the impact of the minimum support on the overall performance of the Apriori. According to [10], when all the other parameters remain the same, including the minimum support, increasing only the number of attributes may result in decreased execution times since the average support for an attribute is lower as the number of attributes grows. This resulted in fewer itemsets and, hence, in faster execution times.

The conclusions that can be drawn after the evaluation of the Apriori algorithm can be summarized as follows. The energy consumption and, thus, the execution time of the algorithm increase linearly with both the number of instances and the number of attributes. In particular, the values in correspondence of which the curve becomes substantially linear are 8 attributes and 42,930 instances (3.2 MB data set size). Just from this data set size, others parameters like the minimum confidence and the minimum support contribute to a lesser extent to the increase of the energy consumption.

4.3 J48 Performance

In this section we illustrate the results of the study of the energy consumption behavior of the J48 algorithm over the Covertype data set. The data set size is 14.5 MB with 114,556 instances and 55 attributes.

The computational complexity of C4.5 is $O(mn^2)$ where m is the number of instances and n the number of attributes. Analogously to what has been said for the Apriori, choosing only those parameters, as indicated in the formula of computational complexity of J48, may not be sufficient to fully evaluate the energy behavior of the algorithm. For this reason, in the following we briefly outline how the C4.5 algorithm is implemented by J48 works.

The C4.5 decision tree classifier algorithm builds a classification tree following two main steps. The first step is to *grow the tree* using a set of data, referred to as the training set. The second step is to *prune the tree* to obtain a smaller tree. The aim is to select the pruned subtree that has the lowest true error rate. C4.5 is based on reduced-error pruning. Pruning the decision trees is, thus, a fundamental step in optimizing the computational efficiency as well as classification accuracy of a classifier. Applying pruning methods to a tree usually results in reducing the size of the tree to avoid unnecessary complexity. Two pruning methods were considered for our experimental study of the J48 algorithm: the *post-pruning* method, and the *online pruning* method. Each method has strengths and weaknesses, the main differences concern the timing of implementation, and the parameters they consider for node removal.

Post-pruning is implemented on a fully induced decision tree removing statistically insignificant nodes. At each node in a tree it is possible to establish the number of instances that are misclassified on a training set by propagating errors upwards from leaf nodes. This computation is performed for all nodes in the tree and the one which has the highest reduced-error rate is pruned. The procedure is then iterated over the just pruned tree until there is no possible reduction in error rate at any node.

Online-pruning differs from post-pruning in that it operates on the decision tree while is being induced. To create the decision tree, the algorithm divides the data set on the attributes that provide the most information gain about the class label; this dividing factor serves as the deciding attribute for each test node. This process continues throughout the creation of the entire tree.

To assess the performance of both pruning methods on the sample domain, a series of trials is performed through the decision tree created by the J48 algorithm. We tested the J48 classifier with confidence factor ranging from 0.15 to 0.75 over a training set of 6,341 instances and 55 attributes. The number of minimum instances per node was set to 2. Figure 11 shows that the energy consumption grows with the confidence factor. This is because the lower the

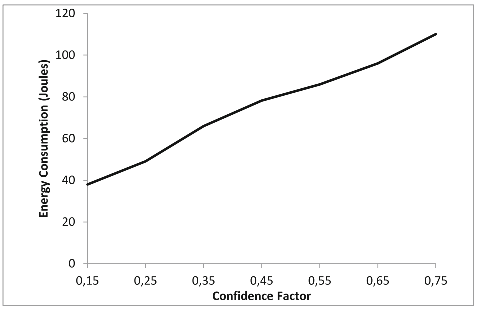


Fig. 11. Energy consumption of J48 w.r.t the confidence factor.

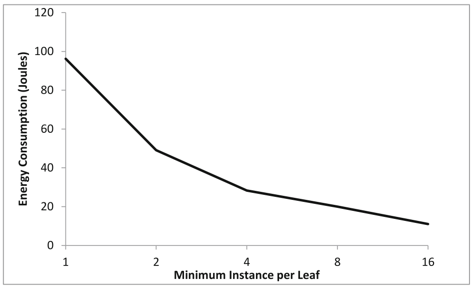


Fig. 12. Energy consumption of J48 w.r.t the minimum instance per leaf.

confidence factor, the higher the chance of pruning and, thus, the lower the tree size. That reduction of tree size leads to faster execution times and, thus, results in a decrease of energy consumption.

For online pruning tests, we tuned the minimum instance per leaf node values ranging from 1 to 16 over 6,341 instances and 55 attributes. Figure 12 shows that by increasing the minimum instance, the energy consumption decreases because the size of the decision tree is reduced.

According to the above discussion, by appropriately tuning C4.5 parameters as the confidence factor, smaller trees can be obtained. Due to the small tree size, the impact of the attributes on the computational complexity of the algorithm is limited as can be argued from the following experiment where the energy consumption of J48 is investigated respect to the number of instances and attributes. Once again we partitioned the original data set into smaller sets of 0.2, 0.4, 0.8 and 1.6 MB corresponding to 1,620, 3,240, 6,480, 11,340 instances, respectively. In each run we varied the number of attributes considering the following representative numbers: 8, 16, 38 and 55 attributes. Figure 13 shows that the energy consumption increases linearly with both the number of instances and the number of attributes. In particular, in correspondence of 1,620 instances the difference among the curves associated to the different number of attributes is

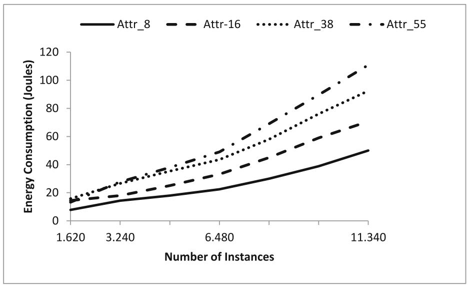


Fig. 13. Energy consumption of J48 w.r.t the number of instance for different number of attributes.

rather limited except for the 8 attributes case that shows a smaller consumption. As the number of instances increases the distance between the curves associated with the different attributes became clearer. Specifically, after 3,240 instances the curves are clearly distinguished from one another. This is particularly evident for 6,480 instances and from this value the energy consumption goes from a sub-linear trend in the number of attributes in a clearly linear trend. This result confirms, as observed also for Apriori, that for small data sets and for small decision trees the computational complexity of the algorithm is lower than the expected one. Decreasing the confidence factor up to ensure an error rate which does not harm the accuracy of the classifier, greatly reduces the computational complexity of the algorithm that is no longer quadratic in the number of attributes but linear in the number of attributes and instances.

5 Conclusion

The development of software frameworks for running data mining tasks on mobile devices will allow to exploit such devices for novel data analysis applications. Handling the energy efficiency issue is a significant contribution for making mobile devices effective platforms for supporting complex applications in nomadic scenarios.

In this paper, we presented a research work aiming to experimentally quantify the performance of representative data mining algorithms from the energy consumption perspective. Our study revealed that by appropriate tuning of a few parameters (e.g., data set size, number of attributes, size of produced results) those algorithms can be efficiently executed on mobile devices.

Acknowledgment. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

References

1. Kargupta, H., Park, B., Pitties, S., Liu, L., Kushraj, D., Sarkar, K.: Mobimine: monitoring the stock marked from a PDA. ACM SIGKDD Explorations **3**(2), 37–46 (2002)
2. Kargupta, H., Bhargava, R., Liu, K., Powers, M., Blair, P., Bushra, S., Dull, J.: VEDAS: a mobile and distributed data stream mining system for realtime vehicle monitoring. In: SIAM Data Mining Conference (2003)
3. Comito, C., Talia, D., Trunfio, P.: An energy-aware clustering scheme for mobile applications. In: IEEE Scalcom'11, pp. 15–22 (2011)
4. Comito, C., Talia, D., Trunfio, P.: An energy aware framework for mobile data mining. In: Zomaya, A., Choon Lee, Y. (eds.) Energy Efficient Distributed Computing Systems, Chapter 23. Wiley-IEEE Computer Society Press, New York (2012)
5. Bhargava, R., Kargupta, H., Powers, M.: Energy consumption in data analysis for on-board and distributed applications. In: ICML 03 Workshop on Machine Learning Technologies for Autonomous Space Applications (2003)

6. Witten, H., Frank, E.: Data Mining: Practical Machine Learning Tools with Java Implementations. Morgan Kaufmann, San Francisco (2000)
7. http://www.android.com/
8. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. Morgan Kaufmann Publishers, San Mateo (2000)
9. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, pp. 281–297 (1967)
10. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules in large databases. In: Proceedings of the 20th International Conference on Very Large Data Bases, pp. 487–499 (1994)
11. Agarwal, R., Mannila, H., Srikant, R., Toivonen, H., Verkamo, A.: Fast discovery of association rules. Advances in Knowledge Discovery and Data Mining, pp. 307–328. MIT Press, Cambridge (1996)
12. Quinlan, J.R.: C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo (1993)
13. http://kdd.ics.uci.edu/databases/

Snooze: An Autonomic and Energy-Efficient Management System for Private Clouds

Matthieu Simonin[1], Eugen Feller[1], Anne-Cécile Orgerie[2](✉), Yvon Jégou[1], and Christine Morin[1]

[1] INRIA, Myriads Team, IRISA, Rennes, France
{matthieu.simonin, eugen.feller, yvon.jegou, christine.morin}@inria.fr
[2] CNRS, Myriads Team, IRISA, Rennes, France
anne-cecile.orgerie@irisa.fr

Abstract. Snooze is an open-source scalable, autonomic, and energy-efficient virtual machine (VM) management framework for private clouds. It allows users to build compute infrastructures from virtualized resources. Particularly, once installed and configured it allows its users to submit and control the life-cycle of a large number of VMs. For scalability, the system relies on a self-organizing and healing hierarchical architecture. Moreover, it performs energy-efficient distributed VM management. Therefore, it implements features to monitor and estimate VM resource utilization (CPU, memory, network Rx, network Tx), detect and resolve overload/underload situations, perform dynamic VM consolidation through live migration, and finally power management to save energy. Last but not least, it integrates a generic scheduler which allows to implement any VM placement algorithm. This demo will expose the Snooze's main properties (scalability, energy-efficiency, autonomy, and fault-tolerance) through its graphical interface.

Keywords: Cloud computing · Virtualized clusters · Energy-efficiency · Autonomic resource management

1 Introduction

The ever growing appetite of new applications for Cloud resources leads to an unprecedented electricity bill for providers. More than the financial cost, the environmental cost is worrying as it threatens the Cloud expansion. It has indeed become a major brake to the deployment of new infrastructures. For instance, in 2010, Google used about 900,000 servers that consumed around 2 billions kWh [1]. One can wonder whether all this energy was necessary. Indeed, Cloud systems are often facing highly variable loads, but they are sized to face the peak power load and their resources are most of the time powered-on even when idle mainly for the sake of reactivity [2]. As current server architectures are not power proportional (i.e. their power consumption is not proportional to the load they are facing) [3], and as their idle consumption (i.e. their power consumption when they are on but idle) is high compared to their peak power consumption, huge

DOI: 10.1007/978-3-642-40517-4_10,

amounts of energy are lost when powered-on servers are idle and this happens often.

In this demo, we will present Snooze [4,5], an autonomic and energy-efficient management system for private clouds. Benefiting from its hierarchical self-configuring architecture, Snooze embeds different autonomic energy-efficient management mechanisms:

- switch off of idle servers,
- energy-efficient VM placement,
- server underload detection and mitigation using live migration to unbalance the load and to increase the number of unused resources,
- periodic VM consolidation.

2 Snooze Architecture

Previous work has shown that hierarchical architectures can greatly contribute to system scalability [6]. Following this principle of splitting the knowledge among independent managers, Snooze is based on an autonomic hierarchical architecture shown in Fig. 1.

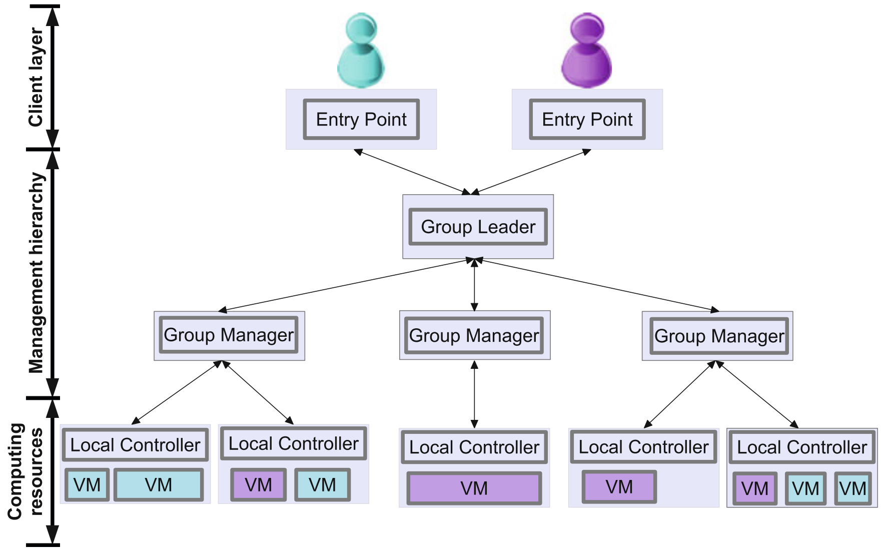


Fig. 1. Snooze architecture

Each server (i.e. physical node) is managed by a *Local Controller* (LC) which monitors the VMs of the node and enforces the VM life-cycle and node management commands coming from higher levels of the hierarchy. A *Group Manager* (GM) is in charge of managing a subset of LCs. It integrates energy-management mechanisms such as overload / underload mitigation and VM consolidation policies. A *Group Leader* (GL) manages the GMs, it assigns the LCs to GMs, and it deals with clients VM submission requests.

On Fig. 1, servers are represented by light gray rectangles. On this example, we chose to have three GMs but this number is defined by the administrator depending on the desired level of redundancy (for fault-tolerance and performance). The main roles and associated functionalities of each component of Snooze's hierarchy are summarized in Table 1.

Table 1. Overview of Snooze hierarchy components

Components	**Roles**	**Functionalities**
Entry point	user interface	availability
Group Leader	deals with client requests assigns GM to LCs dispatches VMs among GM	load-balancing
Group Manager	places VMs overload/underload mitigation VM consolidation (migration plan)	scalability self-optimization energy-efficiency
Local Controller	monitors VMs enforces VM states (start, migration, ...) power off resources overload/underload detection	self-configuration fault-tolerance
Virtual Machines	run applications	user-friendly

Snooze is written in Java. It currently comprises 15,000 lines of highly modular code. It is distributed in open-source under the GPL v2 license [7]. It relies on the *libvirt* library which provides a uniform interface to most of the modern hypervisors [8].

3 Demonstration Scenarios

Snooze has been validated on Grid'5000, the French experimental test-bed to support experiment-driven research in all areas of computer science related to parallel, large-scale or distributed computing [9]. Grid'5000 comprises 6,500 cores geographically distributed in 10 sites linked with a dedicated gigabit network. This demo will be performed on this platform.

In particular in this demo, we will show the main properties of Snooze: the periodic VM consolidation, the power down of unused nodes and their boot when they are needed, the deployment of VM over distant sites, the reconfiguration, fault-tolerance and self-healing mechanisms. Screencasts of the graphical user interface (GUI) of Snooze have been made to illustrate this demo and will be utilized in case of network connection issues during the live demo. The Snooze GUI shows the hierarchy in real-time (with a configurable refreshment period).

During this demo, the following scenarios will be presented:

– a typical Snooze deployment using 50 nodes on 3 different sites of Grid'5000 (and thus on heterogeneous nodes);

- the Snooze energy management in action shutting down unused nodes;
- the wake-up of sleeping nodes when new VMs need to be deployed;
- the on-demand migration mechanism of all the VMs managed by one local controller;
- the periodic VM consolidation for energy saving purposes which migrates VMs among distant sites;
- the self-healing mechanisms and autonomic reconfiguration when a Group Manager or the Group Leader crashes.

4 Conclusion

Snooze [4,5] is a scalable, autonomic, and energy-aware virtual machine management framework. Its name stands for "take a nap" and was selected to emphasize the energy management features of the system, such as the autonomic powering off policy for unused resources.

Snooze [7] is open-source and can thus be used by any research team dealing with Cloud computing, energy efficiency, and resource management in large-scale distributed systems. This demo will also point out the easiness of deployment and management of virtualized clusters provided by Snooze.

Acknowledgement. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

References

1. Koomey, J.: Growth in data center electricity use 2005 to 2010. Technical report. Analytics Press (2011)
2. The Green Grid: Unused servers survey results analysis. Technical report (2010)
3. Barroso, L., Holzle, U.: The case for energy-proportional computing. Computer **40**(12), 33–37 (2007)
4. Feller, E., Rohr, C., Margery, D., Morin, C.: Energy management in IaaS clouds: a Holistic Approach. In: CLOUD: IEEE International Conference on Cloud Computing (2012)
5. Feller, E., Rilling, L., Morin, C.: Snooze : a scalable and autonomic virtual machine management framework for private clouds. In: CCGrid: IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing (2012)
6. Perera, S., Gannon, D.: Enforcing user-defined management logic in large scale systems. In: IEEE Congress on Services, pp. 243–250 (2009)
7. Snooze: http://snooze.inria.fr. GPL v2 licence (2012)
8. Red Hat: libvirt: the virtualization API. http://libvirt.org/ (2012)
9. Bolze, R., Cappello, F., Caron, E., Daydé, M., Desprez, F., Jeannot, E., Jégou, Y., Lanteri, S., Leduc, J., Melab, N., Mornet, G., Namyst, R., Primet, P., Quetier, B., Richard, O., El-Ghazali, T., Touche, I.: Grid'5000: a large scale and highly reconfigurable experimental grid testbed. Int. J. High Perform. Comput. Appl. **20**(4), 481–494 (2006)

DCworms - A Tool for Simulation of Energy Efficiency in Data Centers

Wojciech Piatek(✉)

Poznan Supercomputing and Networking Center, Poznan, Poland
piatek@man.poznan.pl

Abstract. The focus of this paper is a Data Center Workload and Resource Management Simulator (DCworms) which enables modeling and simulation of data centers to estimate their performance, energy consumption, and energy-efficiency metrics for diverse workloads and management policies. We present its functionality in terms of energy-efficiency modeling and methods of power usage estimations available in the simulator. Moreover, we show an application of DCworms within the CoolEmAll project that addresses the important problem of data center energy efficiency.

Keywords: Simulations · Workload and resource modeling · Energy-efficiency · Green data centers

1 Introduction

Rising popularity of large-scale computing infrastructures such as grids and clouds caused quick development of data centers. Nowadays, data centers are responsible for around 2% of the global energy consumption making it equal to the demand of aviation industry [4]. Moreover, in many current data centers the actual IT equipment uses only half of the total energy whereas most of the remaining part is required for cooling and air movement resulting in poor Power Usage Effectiveness (PUE) [6] values. Thus, issues related to cooling, heat transfer, and IT infrastructure location are more and more carefully studied during planning and operation of data centers. For these reasons many efforts were undertaken to measure and study energy efficiency of data centers. In order to optimize a design or configuration of data center we need a thorough study using appropriate metrics and tools evaluating how much computation or data processing can be done within given power and energy budget and how it affects temperatures, heat transfers, and air flows within data center. Therefore, there is a need for simulation tools and models that approach the problem from a perspective of end users and take into account all the factors that are critical to understanding and improving the energy efficiency of data centers, in particular, hardware characteristics, applications, management policies, and cooling. To address these issues, the CoolEmAll project aims at improving energy-efficiency of data centers by providing, among others, a set of data center simulation and visualization tools [1,7]. One of the tools developed within the CoolEmAll project

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 118–124, 2013.
DOI: 10.1007/978-3-642-40517-4_11,

is the Data Center Workload and Resource Management Simulator (DCworms) which enables modeling and simulation of computing infrastructures to estimate their performance, energy consumption, and energy-efficiency metrics for diverse workloads and management policies. More to the point, the presented paper will concentrate on the DCworms, its functionality in terms of energy and power-aware simulations and its application in the CoolEmAll project.

2 DCworms

Data Center Workload and Resource Management Simulator (DCworms) is the event-driven simulator written in Java. It supports studies of dynamic states of IT infrastructures, like power consumption, air throughput and thermal distribution, with respect to the various workload and application profiles, resource models and energy-aware resource management policies. Figure 1 presents the general concept and architecture of the simulator.

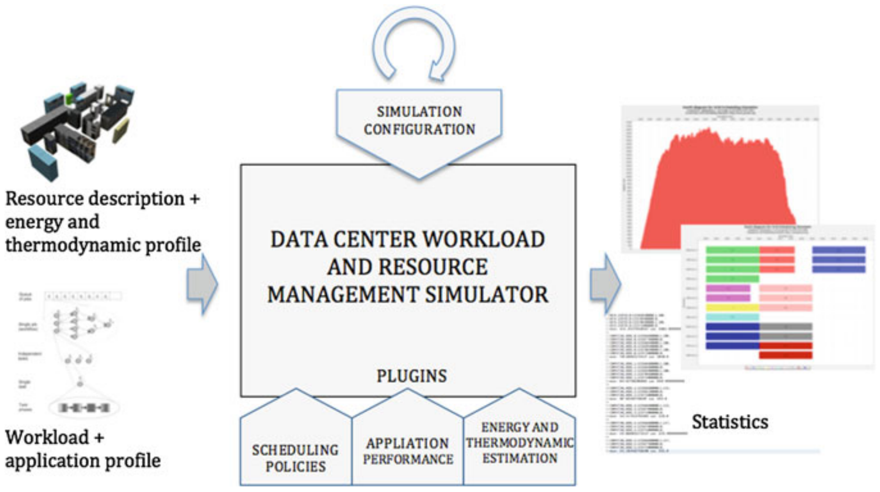

SIMULATION CONFIGURATION
Resource description + energy and thermodynamic profile
Workload + application profile
DATA CENTER WORKLOAD AND RESOURCE MANAGEMENT SIMULATOR
PLUGINS
SCHEDULING POLICIES
APPLIATION PERFORMANCE
ENERGY AND THERMODYNAMIC ESTIMATION
Statistics

Fig. 1. DCworms concept and architecture

In general, input data for the DCWoRMS consist of workload and resources descriptions. They can be provided by the user, read from real traces or generated using the generator module. However, the key elements of the presented architecture are plugins. They allow a researcher to configure and adapt the simulation framework to his/her experiment starting from modeling job performance, through method of calculating energy estimation up to implementation of scheduling policies. Each plugin can be implemented independently and plugged into a specific experiment. Results of experiments are collected, aggregated, and visualized using the statistics tool. Due to a modular and plug-able architecture

DCworms enables adapting it to specific resource management problems and users requirements.

Experiments performed using a workload simulator require a description of the workload itself and applications that will be scheduled during the simulation. As a basic description, DCworms uses files in the Standard Workload Format (SWF) [8]. In addition, some more detailed specification of an application can be included in an additional XML file that provides the scheduler with more detailed information about application profiles and task requirements. To this end, DCworms follows the DNA approach proposed in [3]. This form of representation allows users to define a wide range of workloads: HPC (long jobs, computational-intensive, hard to migrate) or virtualization (short requests) typical for cloud computing environments. The resource description provides a structure and parameters of available resources. Flexible resource definition allows modeling various computing entities consisting of compute nodes, processors and cores. In addition, detailed location of the given resources can be provided in order to group them and form physical structures such as racks and containers. Each of the components may be described by different parameters specifying available memory, storage capabilities, processor speed etc. Moreover, with every component, dedicated profiles can be associated that determines, among others, power, thermal and air throughput properties. These profiles allow to specify the energy characteristics of IT infrastructure. In particular, power profiles allow describing the power usage of resources. Users may provide additional information about power states which are supported by the resources, amounts of energy consumed in these states, and other information essential to calculate the total energy consumed by the resource during runtime. Air throughput profile enables specification of the airflow (related, for example, to the air cooling facilities) depending on the state of resource, where the state include resource power state (in particular if it's on or off) but also its temperature. Finally, the primary goal of thermal profile is to express dependency between power usage and the temperature.

In order to perform comprehensive simulations, including evaluation of workload/resource management policies as well as power and thermodynamic models, DCworms provides dedicated interfaces to incorporate them within the simulation.

Within the scope of energy-efficiency simulation, DCworms benefits from the power and thermodynamic profiles defined within the resource description. Based on this data, it is able to emulate the behavior of the real computing resources. To this end simulator contains a predefined models that include methods to calculate power usage of resources, system air throughput and perform rough estimations of temperatures, too. These models are realized in form of easy to use or exchange plugins that may be plugged into each resource level defined within the evaluate resource architecture. Thus, user can easily develop new models or modify existing ones. By the means of the resource profiles, it is possible to introduce energy related values (derived both from measurements or manufacturers specification) that allows optimizing the simulation models with respect to the real-world data. The main goal of the power consumption model is to simulate the energy usage of the computing resources. Energy estimation plugin can calculate the power consumption based on current resource power state, resource utilization and taking into account the differences in the amount of energy required for executing various types of applications. Thus, it is possible to evaluate various power consumption models that follow changes of resource power states, distinguish between the amount of energy used by idle processors and processors at full load or take into the impact of application type on utilization of computing system components. Air throughput models allow describing the resulting air throughput of the computing system components like cabinets or server fans. Default air flow estimations are based on detailed information about the involved resources, including changes in their air throughput states. Finally, thermal models provide means to introduce temperature sensors simulation. In this way, users have means to approximately predict the temperature of the simulated objects.

DCworms is delivered with a set of resource management policies that can be easily used within the simulation environment. Within the workload management plugin, DCworms provides access to the profiles data, which allows acquiring detailed information concerning current system state. Moreover, it is possible to perform various operations on the given resources, including dynamically changing the frequency level of a single processor, turning off unused resources and managing fan working states. These policies may include a wide spectrum of energy-aware strategies such as workload consolidation/migration, dynamic switching off nodes, Dynamic Voltage and Frequency Scaling and thermal-aware methods. In addition to typical approaches minimizing energy consumption, policies that prevent too high temperatures in the presence of limited cooling (or no cooling) may also be analyzed. Moreover, apart from the set of predefined strategies, new approaches can easily be applied and examined.

In terms of applications behavior modeling, DCworms provides means to include complex and specific application performance models during simulations. To this end, DCworms is supported with a dedicated module, which based on the application profile is able to estimate its execution time. Using this functionality the impact of architectures of the underlying systems, such as multi-core processors, or virtualization overheads on the final performance of applications can be taken into account.

The output of the simulation consists of a set of textual and graphical statistics concerning resource utilization, resource energy usage, temperature and air throughput measurements as well as application performance data.To ensure appropriate level of details each change of the resource component state and each change of the measured values are logged along with the timestamp. These outcomes are based on the values calculated by the simulator (in case of the resource load) or are derived from the user estimations returned by the dedicated plugins. As a result, statistics for each resource entity are represented in form of pairs 'timestamp, value' that allows tracking their changes over time. Moreover, for each characteristic the statistical values are also calculated. Finally, all measurements can be visualized in form of the linear charts that facilitate the analysis of large volumes of data.

Results of example experiments performed using DCworms and more detailed description of the simulator can be found in [5].

3 CoolEmAll

The main goal of CoolEmAll is to provide advanced simulation, visualization and decision support tools along with blueprints of computing building blocks for data centers. Once developed, these tools and blueprints should help to minimize the energy consumption, and consequently the CO_2 emissions of data centers. This will be achieved by:

- design of diverse types of computing building blocks well defined by hardware specifications, physical dimensions, and energy efficiency metrics,
- development of simulation, visualization and decision support toolkit (SVD Toolkit) that will enable analysis and optimization of IT infrastructures built of these building blocks.

Both building blocks and the toolkit will take into account aspects that have major impact on actual energy consumption: hardware characteristics, cooling solutions, properties of applications, and workload and resource management policies. To achieve it, the energy efficiency of computing building blocks will be precisely defined by a set of metrics expressing relations between the energy efficiency and essential factors listed above. In addition to common static approaches, the CoolEmAll platform will enable studies of dynamic states of data centers based on changing workloads, management policies, cooling method, and ambient temperature.

4 DCworms in CoolEmAll

As said before, the goal of the SVD toolkit is to enable interactive analysis of data centers with respect to various important aspects that may affect their energy-efficiency. One of these aspects concerns workload and resource management policies. The main aim of the workload simulation stage is to enable studies of dynamic states of IT infrastructures, like power consumption and air throughput distribution, on the basis of changing workloads, resource model and energy-aware resource management policies. Workload simulation phase takes into account the specific workload and application characteristics as well as detailed resource parameters. It will benefit from the CoolEmAll benchmarks and classification of applications and workloads. In particular various types of workload, including data center workloads using virtualization and HPC applications, will be considered. The knowledge concerning their performance and properties as well as information about their energy consumption and heat production will be used in simulations to study their impact on thermal issues and energy efficiency. The resource model is based on building blocks description that supports modeling a data center at various granularity levels. Besides defining simulated architecture, it is complemented with resource energy profiles that become an additional criterion in the workload management process. Based on this data workload simulation will support evaluation process of various resource management approaches. The outcome of the workload simulation phase is a distribution of power usage and air throughput for the computing models specified within the SVD Toolkit. These statistics may be analyzed directly by data center designers and administrators and/or provided as an input to the Computational Fluid Dynamics (CFD) simulation phase. The former case allows studying how the above metrics change over time, while the latter harness CFD simulations to identify temperature differences between the computing modules, called hot spots. The goal of this scenario is to visualize the behavior of the temperature distribution within a server room with a number of racks for different types of executed workloads and for various policies used to manage these workloads. The following figure (Fig. 2) shows the 3D model and a temperature distribution for the part of the compute building block used within the CoolEmAll testbed. Additional results of simulation experiments performed with DCworms, based on the CoolEmAll testbed, are provided in [2].

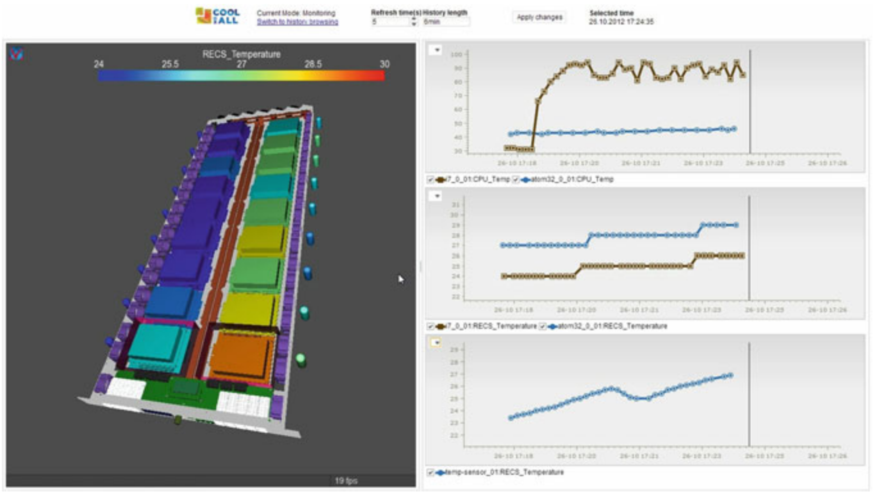

Current Mode: Monitoring
Refresh time(s): 5
History length: 6min
Selected time: 26.10.2012 17:24:35

Fig. 2. 3D model of the CoolEmAll testbed with the corresponding temperature distribution

Acknowledgements. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

References

1. vor dem Berge, M., Da Costa, G., Kopecki, A., Oleksiak, A., Pierson, J.-M., Piontek, T., Volk, E., Wesner, S.: Modeling and simulation of data center energy-efficiency in CoolEmAll. In: Huusko, J., de Meer, H., Klingert, S., Somov, A. (eds.) E2DC 2012. LNCS, vol. 7396, pp. 25–36. Springer, Heidelberg (2012)
2. vor dem Berge, M., Da Costa, G., Jarus, M., Oleksiak, A., Piatek, W., Volk, E.: Modeling data center building blocks for energy-efficiency and thermal simulations. In: 2nd International Workshop on Energy-Efficient Data Centres, Berkeley (2013)
3. Chetsa, G.L.T., Lefevre, L., Pierson, J.-M., Stolf, P., Da Costa, G.: DNA-inspired Scheme for Building the Energy Profile of HPC Systems. In: Huusko, J., de Meer, H., Klingert, S., Somov, A. (eds.) E2DC 2012. LNCS, vol. 7396, pp. 141–152. Springer, Heidelberg (2012)
4. Koomey, J.: Worldwide electricity used in data centers. Environ. Res. Lett. **3**, 034008 (2008)
5. Kurowski, K., Oleksiak, A., Piatek, W., Piontek, T., Przybyszewski, A., Weglarz, J.: DCWoRMS - a tool for simulation of energy efficiency in distributed computing infrastructures, Simulation Modelling Practice and Theory (2013) (in revision)
6. The Green Grid Data Center Power Efficiency Metrics: PUE and DCiE. http://www.thegreengrid.org/Global/Content/white-papers/The-Green-Grid-Data-Center-Power-Efficiency-Metrics-PUE-and-DCiE
7. The CoolEmAll project website. http://coolemall.eu
8. ParallelWorkload Archive. http://www.cs.huji.ac.il/labs/parallel/workload/

Energy Efficiency in Secure and Dynamic Cloud Storage

Adilet Kachkeev(✉), Ertem Esiner, Alptekin Küpçü, and Öznur Özkasap

Department of Computer Science and Engineering, Koç University, İstanbul, Turkey
{akachkeev, eesiner, akupcu, oozkasap}@ku.edu.tr

Abstract. The popularity of the cloud storage systems has brought a number of challenges. Two of them are data integrity and energy efficiency. There are many proposed static solutions to prove the integrity of a file. For the dynamic case, Rank-Based Authenticated Skip list (RBASL) has been presented. It provides the update operations with logarithmic complexity. However, an RBASL expects the block size to be fixed. In a realistic scenario, where the updates can be of variable size, an RBASL performs $O(n)$ update operations on the data structure when a change in the file occurs, where n is the number of blocks. To overcome this problem, we propose Flexible-length based authenticated skip list (FlexList) to make $O(u)$ update operations where u is the number of the update operations. Moreover, we developed an algorithm to carry out multiple challenges at once. We have tested our algorithm, and the results show time and energy efficiencies of 60%, 45%, 35% and 20% for file sizes 4MB, 40MB, 400MB, and 4GB respectively.

Keywords: Energy efficiency · Cloud storage · Skip list · Provable data possession

1 Introduction

Cloud data storage systems have become popular in recent years both in academia [1,3,6,7,10,13] and industry (e.g., Sky Drive, Google Drive, Amazon S3), where energy efficiency and proving data integrity have become important challenges [4]. These systems have two entities, a client and a server. A client sends her data to the server - data storage provider (third party), which promises to store the data intact and provide its availability. However, the server can be malicious, and even if the server is trustworthy, there may be hardware or software related failures that may cause data corruption or loss. Therefore, the client should be able to efficiently (in terms of energy and time) and securely check the integrity of her data without downloading the whole file [1].

One of the first proposed models with provable data integrity is *Provable Data Possession* (PDP) [1]. The client, in this model, has the ability to challenge the server on randomly chosen blocks, the server sends a proof, and she can verify the data integrity through that proof. PDP and other related schemes are applicable to the static cases, if the blockwise update operations (insert, remove, modify) are possible, they demonstrate poor performance [1,6,10,13]. The static scenario can be used in some systems (e.g., archival storage at the libraries), but many other applications may necessitate a dynamic scenario, where the client can interact with her data in a read/write manner, while preserving the data possession guarantees. Ateniese et al. [3] proposed Scalable PDP, where the client has a pre-determined number of a limited set of operations. Erway et al. [7] proposed a model called *Dynamic Provable Data Possession*, which not only extends the PDP model, but also provides a dynamic solution. However, an underlying authenticated data structure based on a skip list [12] is needed for the implementation of the DPDP scheme.

For the dynamic scenario, Erway et al. [7] introduced the new data structure *rank-based authenticated skip list (RBASL)*, which is a special type of authenticated skip list [9] to be used in DPDP. In this model, the client preprocesses the file and stores meta data to verify the later proofs from the server. Then she outsources the file to the server. At any time, she can challenge some blocks to check the integrity of her file. Upon such request, the server prepares a proof for the challenged blocks. In opposition to an authenticated skip list, where a search is done using a key, in an RBASL one can search with indices of the blocks. This feature gives the opportunity to efficiently check the integrity of the data using block indices. Authenticated ranks are used as a search key in an RBASL. Each node has a *rank*, indicating the count of the nodes at leaf-level that are reachable from that particular node. Leaf-level nodes, having no *after* links, have a rank of 1.

In a realistic scenario, the client may want to alter some part of a particular block, not the whole. It can be problematic to perform in an RBASL. Modification of a particular block in an RBASL may cause the modifications in all consequent blocks as well. Therefore, it is subject to $O(n)$ modifications for DPDP and PDP, which is inefficient in terms of time and energy. We propose a new data structure called FlexList, which is based on an authenticated skip list. It performs dynamic operations (modify, insert, remove) for cloud data storage, while having $O(1)$ variable block sized updates. Moreover, the client has a capability to challenge multiple blocks at once and the server using the multi-proof algorithm prepares the proof for the client.

Our main contribution is as follows:

- The client can challenge the server for multiple blocks using authenticated skip lists, rank-based authenticated skip lists and FlexLists. Our algorithm provides an *optimal* proof, without any repeated items. The current experimental results show time and energy efficiencies of 60%, 45%, 35% and 20% for file sizes 4MB, 40MB, 400MB, and 4GB respectively.

2 Secure and Dynamic Cloud Storage Using FlexList

2.1 FlexList

A fixed block size is suitable for an RBASL, since a search (and other methods) by byte index of the data is not possible with rank information. A FlexList, unlike an RBASL, supports the variable size of a block. Due to the problem of providing variable block sized operations with an RBASL, we present a FlexList, which overcomes the problem and serves as an underlying data structure in our cloud storage system. A FlexList stores, at every node, the total number of *bytes* that can be reached from that node, instead of the number of blocks reachable from it. The *rank* of each leaf-level node is computed as the sum of the *length* of its data and the *rank* of its *after* node (0 if *null*). The *rank* for every non-leaf node is computed as the summation of the *ranks* of its **below** and *after* links [8].

2.2 FlexDPDP

FlexList is employed as an authenticated data structure in our secure and dynamic cloud storage systems, which supports energy efficient data integrity checking. We define a FlexDPDP as a DPDP scheme with FlexList. In our system we have two main parties: a client and a server. The server provides storage space for the client's file. An RBASL can be constructed on the top of the file as shown by Erway et al. [7]. A FlexList, in contrast to an RBASL, can search and reach the data bytes easily, even though the data blocks are of variable sizes. Therefore, a FlexList can perform a variable sized update of length $O(u)$ in $O(u)$ operations. However, an RBASL will need $O(n)$ operations, where n is the number of the blocks in the file. A FlexList represents file blocks as the leaves. So the search path for particular block is the proof membership (i.e., integrity) from this leaf node to the root. We developed an optimized proof generation algorithm, which handles the multiple block challenge at once [8].

The FlexDPDP scheme employs *homomorphic verifiable tags* (as DPDP), so that a number of tags can be combined to obtain a single tag that corresponds to the combined blocks [2]. Small size of tags compared to data blocks enables storage in memory. The authenticity of the skip list gives guarantees for the integrity of tags, and tags protect the integrity of the data blocks.

2.3 Multi-Proof Generation

The client server system starts with the client preprocessing her data, where a FlexList on the file is created and tags are calculated for each block. Then, the client send the random seed to the server and the file itself. The server using the seed can create the identical FlexList using the data blocks and tags (sent by the client). The server sends the hash value of the root of the FlexList for the client to verify the correct construction of the FlexList. After the successful verification, the client can safely delete the file and keep the hash value of the root as the meta data. At any time later, the client can send a random seed to the server to challenge a number of blocks. The server using the seed creates the challenges, runs the *genMultiProof* algorithm and returns proof generated by this algorithm to the client. She can verify the proof using the meta data and the verification algorithm.

genMultiProof: The proof generation algorithm is run by the server, upon the receipt of the random seed from the client. The server first generates a predetermined number of challenges, and random values accordingly. Then, the server runs the *genMultiProof* algorithm to obtain the proof, file blocks and tags for the challenged indices. The algorithm traverses to the leaf-level nodes holding the challenged blocks. Along the traverse path, it stores visited nodes. We have observed that the regular search for each challenged block is inefficient. Since the regular search always starts from the root, there are a lot of repeated nodes in the proof. To handle this problem, we save the states at each intersection point. A node is an intersection point of proof paths of two indices when the first index can be found following the **below** link and the second index is detected by following the *after* link. Note that all the challenges (indices) are in ascending order. In our *optimal* proof, we visit and take information stored for each node on the proof path only once [8]. Therefore, the algorithm achieves significant gains in terms of time and energy.

3 Evaluation

We developed a prototype implementation of an optimized FlexList and used it in our FlexDPDP scheme. C++ is used as the programming language and some of the methods from the *Cashlib* library [5,11] are employed. The experiment was conducted on a 64-bit machine with a 2.4GHz Intel 4 core CPU (only one core is active), 4GB main memory and 8MB L2 cache, running Ubuntu 12.10. As security parameters, we used 1024-bit RSA modulus, 80-bit random numbers, and SHA-1 hash function, overall resulting in an expected security of 80-bits. Watts up Pro meter was used for the energy efficiency tests. It measures the total energy consumption of the connected device and displays this information. We measured the average energy consumption while the experiment was running. Then, we measured the average energy cost for the idle time, while no tests were taking place. The difference between these two measurements were used in the calculation of the results. Energy consumption and time (CPU) ratio results are close in our experiment. Since there was no I/O delay in the test due to disk access, we argue that the energy efficiency of our *genMultiProof* algorithm is directly impacted by the CPU time. The test is the average of 10 runs.

We have tested our *genMultiProof* algorithm, which is used to accumulate the proof along with the FlexList for the received challenge request from the client. The ratio graph for the *genMultiProof* algorithm is shown in Fig. 1. We had different file size scenarios: starting a file size from 4MB to 4GB (all with a block size of 2KB). In every scenario we had the same challenge size of 460, which is sufficient for high probability (constitutes to 99% with the assumption that 1% of the file altered) of catching the cheating server. The time/energy ratio is a division of the time/energy spent for the operation of 460 challenges (one by one) to the time/energy spent for the *genMultiProof* algorithm. Even though the results show a decline in the time and energy ratio while the file size grows, the efficiency of the algorithm is still satisfactory. The algorithm gets its advantage through the minimization of the proof size, therefore no repeated proof nodes for the same node are created. Once created, the proof node is used

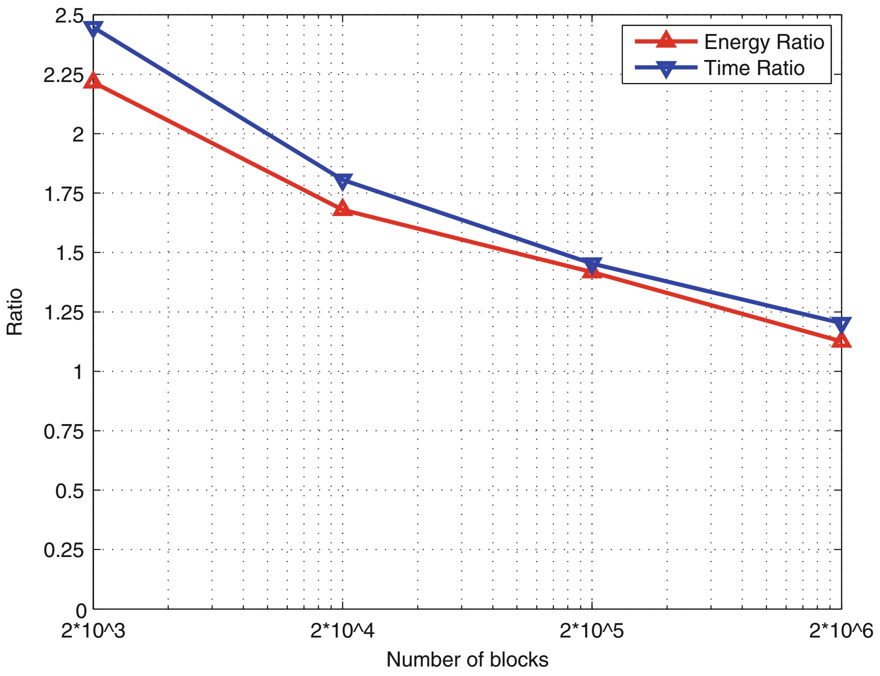


Fig. 1. Ratio graph on *genMultiProof* algorithm.

as a common proof node for other nodes as well. As the file size grows, the ratio on the number of common proof nodes to the total proof size decreases, since we have the constant number of challenges as 460. Note that the maximum efficiency of the *genMultiProof* algorithm is reached when the challenged blocks are near each other. Nevertheless, the graph clearly shows the efficiency gains in terms of energy and time for sufficiently large file sizes. So for the file of size 4MB, 40MB, 400MB and 4GB, we have time and energy gains of 60%, 45%, 35% and 20% respectively.

4 Conclusion and Future Work

With the emergence of the distributed and cloud storage services, energy efficiency has become one of the important challenges [4]. Early works have shown that the static solutions with optimal complexity [1,13], and the dynamic solutions with logarithmic complexity [7] are within reach. However, the DPDP [7] solution is not applicable to the realistic scenarios since it supports only fixed block size and therefore lacks flexibility on the data updates, while updates in the realistic scenario are more likely to be of a variable size. We have extended their work in several ways and provided a new data structure (FlexList) and its usage in the cloud data storage. A FlexList efficiently supports the variable block sized multiple updates, and we showed how handling multiple challenges at once greatly improves scalability and energy efficiency. As a part of future work, we plan to further develop FlexDPDP algorithms. Currently, we are working on energy efficient algorithms to create a FlexList from a scratch, perform and verify multiple updates. Subsequently, a P2P model for the FlexDPDP will be investigated and designed. We plan to deploy such a system on PlanetLab and run tests for energy efficiency at both the client and the server.

Acknowledgements. Our work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804: "Energy Efficiency in Large Scale Distributed Systems" and Action IC1206, by TÜBİTAK (the Scientific and Technological Research Council of Turkey) under grants 109M761 and 112E115, by Türk Telekom, Inc. under Grant 11315-06, and by Koç Sistem, Inc.

References

1. Ateniese, G., Burns, R., Curtmola, R., Herring, J., Kissner, L., Peterson, Z., Song, D.: Provable data possession at untrusted stores. In: ACM CCS (2007)
2. Ateniese, G., Kamara, S., Katz, J.: Proofs of storage from homomorphic identification protocols. In: Matsui, M. (ed.) ASIACRYPT 2009. LNCS, vol. 5912, pp. 319–333. Springer, Heidelberg (2009)
3. Ateniese, G., Pietro, R.D., Mancini, L.V., Tsudik, G.: Scalable and efficient provable data possession. In: SecureComm (2008)
4. Berl, A., Gelenbe, E., Di Girolamo, M., Giuliani, G., De Meer, H., Dang, M.Q., Pentikousis, K.: Energy-efficient cloud computing. Comput. J. **53**(7), 1045–1051 (2010)
5. Brownie cashlib cryptographic library. http://github.com/brownie/cashlib
6. Dodis, Y., Vadhan, S., Wichs, D.: Proofs of retrievability via hardness amplification. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 109–127. Springer, Heidelberg (2009)
7. Erway, C., Küpçü, A., Papamanthou, C., Tamassia, R.: Dynamic provable data possession. In: ACM CCS (2009)
8. Esiner, E., Kachkeev, A., Küpçü, A., Özkasap, Ö.: Flexlist: optimized skip list for secure cloud storage. Technical Report, Koç University (2013). http://crypto.ku.edu.tr/sites/crypto.ku.edu.tr/files/papers/techreport-flexl
9. Goodrich, M.T., Tamassia, R., Schwerin, A.: Implementation of an authenticated dictionary with skip lists and commutative hashing. In: DARPA (2001)
10. Juels, A., Kaliski, B.S.: PORs: Proofs of retrievability for large files. In: ACM CCS (2007)
11. Meiklejohn, S., Erway, C., Küpçü, A., Hinkle, T., Lysyanskaya, A.: Zkpdl: enabling efficient implementation of zero-knowledge proofs and electronic cash. In: USENIX Security (2010)
12. Pugh W.: Skip lists: a probabilistic alternative to balanced trees. Commun. ACM **33**: 668–679 (1990)
13. Shacham, H., Waters, B.: Compact proofs of retrievability. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 90–107. Springer, Heidelberg (2008)

Part III

HPC Computing

A Holistic Model of the Performance and the Energy-Efficiency of Hypervisors in an HPC Environment

Mateusz Guzek[1](✉), Sébastien Varrette[2], Valentin Plugaru[2], Johnatan E. Pecero[2], and Pascal Bouvry[2]

[1] Interdisciplinary Centre for Security Reliability and Trust, University of Luxembourg, Luxembourg, Luxembourg
[2] Computer Science and Communications (CSC) Research Unit, University of Luxembourg, Luxembourg, Luxembourg
{Mateusz.Guzek, Sébastien.Varrette, Valentin.Plugaru, Johnatan.E.Pecero, Pascal.Bouvry}@uni.lu

Abstract. With a growing concern on the considerable energy consumed by HPC platforms and data centers, research efforts are targeting toward green approaches with higher energy efficiency. In particular, virtualization is emerging as the prominent approach to mutualize the energy consumed by a single server running multiple Virtual Machines (VMs) instances. However, little understanding has been obtained about the potential overhead in energy consumption and the throughput reduction for virtualized servers and/or computing resources, nor if it simply suits an environment as high-demanding as a High Performance Computing (HPC) platform. In this paper, a novel holistic model for the power of HPC node and its eventual virtualization layer is proposed. More importantly, we create and validate an instance of the proposed model using concrete measures taken on the Grid5000 platform. In particular, we use three widespread virtualization frameworks, namely Xen, KVM, and VMware ESXi and compare them with a *baseline* environment running in native mode. The conducted experiments were performed on top of benchmarking tools that reflect an HPC usage, i.e. HPCC, IOZone and Bonnie++. To abstract from the specifics of a single architecture, the benchmarks were run using two different hardware configurations, based on Intel and AMD processors. The benchmark scores are presented for all configurations to highlight their varying performance. The measured data is used to create a statistical holistic model of power of a machine that takes into account impacts of its components utilization metrics, as well as used application, virtualization, and hardware. The purpose of the model is to enable estimation of energy consumption of HPC platforms in areas such as simulation, scheduling or accounting.

Keywords: Energy-efficiency · HPCC · IOZone · Bonnie++ · Xen · KVM · ESXi

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 133–152, 2013.
DOI: 10.1007/978-3-642-40517-4_13,

1 Introduction

With the advent of the Cloud Computing (Cloud Computing (CC)) paradigm, more and more workloads are being moved to virtual environments. Yet the question of whether CC is suitable for High Performance Computing (HPC) workload remain unclear. With a growing concern on the considerable energy consumed by HPC platforms and data centers, having a clear answer to this question becomes more and more crucial.

In this paper, we evaluate and model the overhead induced by several virtualization environments (often called *hypervisors*) at the heart of most if not all CC middlewares. In particular, we analyze in this study the performance and the energy profile of three widespread virtualization frameworks, namely Xen, KVM, and VMware ESXi, running a single VM instance and compare them with a *baseline* environment running in native mode. It is worth to notice that it is quite hard to find in the literature fair comparisons of `all` these hypervisors. For instance, in the few cases where the VMWare suite is involved, the study is generally carried on by the company itself. The experiments presented in this paper were performed on top of benchmarking tools that reflect an HPC usage, i.e. the HPC Challenge (HPCC), IOZone and Bonnie++. They helped to refine a novel holistic model for the power consumption of HPC components which is proposed in this article. Moreover, to abstract from the specifics of a single architecture, the benchmarks were run using two different hardware configurations, based on Intel and AMD processors. In this context, the Grid'5000 platform helped to deploy in a flexible way such heterogeneous configuration and provide a unique environment as close as possible to a real HPC system. To this extent, the work presented in this paper offers an interesting complement to precedent studies, which targeted a similar evaluation, yet limited the analysis to a subset of hypervisors (generally excluding VMWare products) and a fewer number of benchmarks. Our experimental settings reflect the increasing complexity of HPC systems' analysis and management. From the green computing perspective it is crucial to be able to estimate and predict the power and consequently the energy of a data center. Such prediction could be used to optimize the system by assisting scheduling or hypervisor choice, to simulate systems' future behavior or even to account the consumed energy in case of insufficient physical infrastructure. The hardware, configuration and type of processing, has impact on the power consumption of a machine, which is reflected in the novel model. First, the model redefines the structure of computing in a virtualized data center. The classical Task and Machine models are extended by the Configuration layer that represents the chosen middleware. Then, we propose a power model that combines multiple factors, either directly measured utilization metrics or classifiers such as used node, hypervisor or application, consequently calling the model *holistic*. The power modelling is lightweight in terms of creation and usage, as it is based on multiple linear regression. The purpose of power modelling is to validate the theoretical assumption that increasing the amount of information about the node enables better power estimation.

This article is organized as follows: Section 2 presents the background of this study and reviews related work. Then, our novel holistic model is detailed in Sect. 3. Section 4 describes the experimental setup used within this study, in particular we will present the different cutting-edge platforms we compare, together with the benchmark workflow applied to operate these comparisons. Then, Sect. 5 details the experimental results obtained. Finally, Sect. 6 concludes the paper and provides the future directions.

2 Context and Motivations

In essence, Cloud middleware exploit virtualization frameworks that authorize the management and deployment of Viryual Machines (VMs). Whereas our general goal is to model Cloud systems in an HPC context, we present here the first step toward this global modelization focusing on the underlying hypervisor or Virtual Machine Manager. Subsequently, a VM running under a given hypervisor will be called a guest machine. There exist two types of hypervisors (either *native* or *hosted*) yet only the first class (also named bare-metal) presents an interest for the HPC context. This category of hypervisor runs directly on the host's hardware to control the hardware and to manage guest operating systems. A guest operating system thus runs on another level above the hypervisor. Among the many potential approach of this type available today, the virtualization technology of choice for most open platforms over the past 7 years has been the Xen hypervisor [6]. More recently, the Kernel-based Virtual Machine (KVM) [15] and VMWare ESXi [5] have also known a widespread deployment within the HPC community such that we limited our study to those three competitors and decided to place the other frameworks available (such as Microsoft's Hyper-V or OpenVZ) out of the scope of this paper. Table 1 provides a short comparison chart between Xen, KVM and VMWWare ESXi.

2.1 Considered HPC Platforms

To reflect a traditional HPC environment, yet with a high degree of flexibility as regards the deployment process and the fair access to heterogeneous resources,

Table 1. Overview of the considered hypervisors characteristics.

Hypervisor:	**Xen 4.0**	**KVM 0.12**	**ESXi 5.1**
Host architecture	x86, x86-64, ARM	x86, x86-64	x86-64
VT-x/AMD-v	Yes	Yes	Yes
Max Guest CPU	128	64	32
Max. Host memory	1TB	-	2TB
Max. Guest memory	1TB	-	1TB
3D-acceleration	Yes (HVM Guests)	No	Yes
License	GPL	GPL/LGPL	Proprietary

the experiments presented in this paper were carried on the Grid'5000 platform [2]. Grid'5000 is a scientific instrument for the study of large scale parallel and distributed systems. It aims at providing a highly reconfigurable, controllable and monitorable experimental platform to its users. One of the unique features offered by this infrastructure compared to a production cluster is the possibility to provision on demand the Operating System (OS) running on the computing nodes. Designed for scalability and a fast deployment, the underlying software (named Kadeploy) supports a broad range of systems (Linux, Xen, *BSD, etc.) and manages a large catalog of images, most of them user-defined, that can be deployed on any of the reserved nodes of the platform. As we will detail in Sect. 4, we have defined a set of common images and environments that were deployed in two distinct hardware architectures (based on Intel or AMD) on sites that offer the measurement of Power distribution units (PDUs).

2.2 Considered Benchmarks

Several benchmarks that reflect a true HPC usage were selected to compare all of the considered configurations. For reproducibility reasons, all of them are open source and we based our choice on a previous study operated in the context of the FutureGrid[1] platform [17], and a better focus on I/O operation that we consider as under-estimated in too many studies involving virtualization evaluation. We thus arrived to the three benchmarks:

The HPCC Challenge (HPCC) [13], an industry standard suite used to stress the performance of multiple aspects of an HPC system, from the pure computing power to the disk/RAM usage or the network interface efficiency. It also provides reproducible results, at the heart of the ranking proposed in the Top500 project. HPCC basically consists of seven tests: (1) `HPL` (the High-Performance Linpack benchmark), which measures the floating point rate of execution for solving a linear system of equations. (2) `DGEMM` - measures the floating point rate of execution of double precision real matrix-matrix multiplication. (3) `STREAM` - a simple synthetic benchmark program that measures sustainable memory bandwidth (in GB/s) and the corresponding computation rate for simple vector kernel. (4) `PTRANS` (parallel matrix transpose) - exercises the communications where pairs of processors communicate with each other simultaneously. It is a useful test of the total communications capacity of the network. (5) `RandomAccess` - measures the rate of integer random updates of memory (GUPS). (6) `FFT` - measures the floating point rate of execution of double precision complex one-dimensional Discrete Fourier Transform (DFT). (7) Communication bandwidth and latency - a set of tests to measure latency and bandwidth of a number of simultaneous communication patterns.

Bonnie++ [1], a file system benchmarking suite that is aimed at performing a number of simple tests of hard drive and file system performance.

[1] See https://portal.futuregrid.org/.

IOZone [3], a more complete cross-platform suite that generates and measures a variety of file operations. Iozone is useful for performing a broad filesystem analysis of a given computing platform, covering tests for file I/O performances for many operations (Read, write, re-read, re-write, read backwards/strided, mmap etc.)

The results that are obtained from these benchmarks provide an unbiased performance analysis of the hypervisors and thus provide a valid reference for the holistic model proposed in this article.

2.3 Related Work

Different studies describe or apply models for power draw for data centers and HPC centers subsystems. Some modelling research work indicate that server power varies roughly linearly in CPU utilization [10]. Economu et al. [9] study the component-level power breakdown and variation, as well as temporal workload-specific power consumption of a blade server. The authors suggest to consider beside CPU and disk utilization also design properties of the server. Kansal et al. [14] proposed a power meter model for virtual machines, called *Joulemeter*. The model makes use of power models of individual hardware resources; at runtime software components monitor the resource usage of VMs and they convert it to energy usage using the available model. Bohra et al. in [7] proposed *vMeter*, a power modelling technique. The authors observed a correlation between the total system's power consumption and component utilization. They proposed a four-dimensional linear weighted power model for the total power consumed. The components of the model are: performance parameters for CPU, cache, DRAM and disk. The weights of the model are calculated per workflow. The authors refunded the power model by separating the contribution of each active domain in a node, either a VM or domain. Chen et al. [8] present a study in profiling virtual machines with respect to three power metrics: power, power efficiency and energy, under different high performance computing workloads. The authors proposed a linear power model that represents the behavior of a single work node and includes the contribution from individual components. Liu et al. [16] proposed a GreenCloud architecture that utilizes live migration of VMs based on power information of the physical nodes reducing energy consumption for applications running in clouds, specifically for online gaming.

At the level of the pure hypervisor performance evaluation, many studies can be found in the literature that attempt to quantify the overhead induced by the virtualization layer. Yet the focus on HPC workloads is recent as it implies several challenges, from a small system footprint to efficient I/O mechanisms. A first quantitative study was proposed in 2007 by A. Gavrilovska et al. in [4]. While the claimed objective was to present opportunities for HPC platforms and applications to benefit from system virtualization, the practical experimentation identified the two main limitations to be addressed by the hypervisors to be of interest for HPC: I/O operations and adaptation to multi-core systems. While the second point is now circumvented on the considered hypervisor systems, the first one remains challenging. Another study that used to guide not only our benchmarking strategy but also our experimental setup is the evaluation mentioned in the previous section that was performed on the FutureGrid platform [17]. The targeted hypervisors were Xen, KVM, and Virtual Box and a serious performance analysis is proposed, with the conclusion that KVM is the best overall choice for use within HPC Cloud environment. Compared to the above mentioned studies, our contribution in this paper can be summarized in the following elements: (1) a novel holistic model for the power consumption of HPC components, eventually running on top of three widespread virtualization frameworks (Xen, KVM and VMware ESXi), is proposed; (2) the model is refined using the hypervisors in a concrete HPC environment, comparing also the two leading concurrent hardware architecture (Intel and AMD – 84% of the represented processor technologies in the latest Top500 list). Our performance evaluation involves industrial reference benchmarks and does not ignore a measure of the impact on I/O operations, too often ignored in the literature; (3) it is one of the few independent study that takes into consideration not only open-source hypervisors (Xen and KVM) but also a proprietary solution from the leading vendor in the domain i.e. VMWare; (4) the energy-efficiency of the considered configuration is properly modelled and quantified with exact measures offered by the Grid5000 platform.

3 The Holistic System Model

In this section, we introduce a novel model for the performance and energy-efficiency analysis of HPC or CC components. The model is holistic, i.e. it includes all elements important for the performance and power of distributed computing systems, and it relies on classical scheduling models with machines and tasks. Our first contribution is the addition of a machine configuration layer that corresponds to the used middleware. The bottom layer, the *Machine* layer, describes the physical characteristics of the host. Modelling that layer assumes the derivation of the energy model of a machine which is based on the measured utilization of its components or environmental factors (e.g. temperature, supply voltage). In this work we take into account utilization metrics of CPU, Memory, Disk and Networking of a node. This layer describes each machine separately, taking into account their heterogeneity.

The middle *Configuration* layer corresponds to the overhead induced by the software that is used to process tasks. The basic element of this layer is *Container* that represents the used OS, including a potential virtualization technology. In case of virtualized systems there can exist multiple, possibly heterogenous, containers on a single machine.

The top *Task* layer represents the computation or work performed by applications. A *Task* represents a workload processed by an application and its corresponding data. Multiple tasks can be executed simultaneously in a single Container, with the performance depending on the availability of resources.

The second contribution of the proposed model is an extension of the classical definition of resources. Instead of representing resources as discrete entities, the

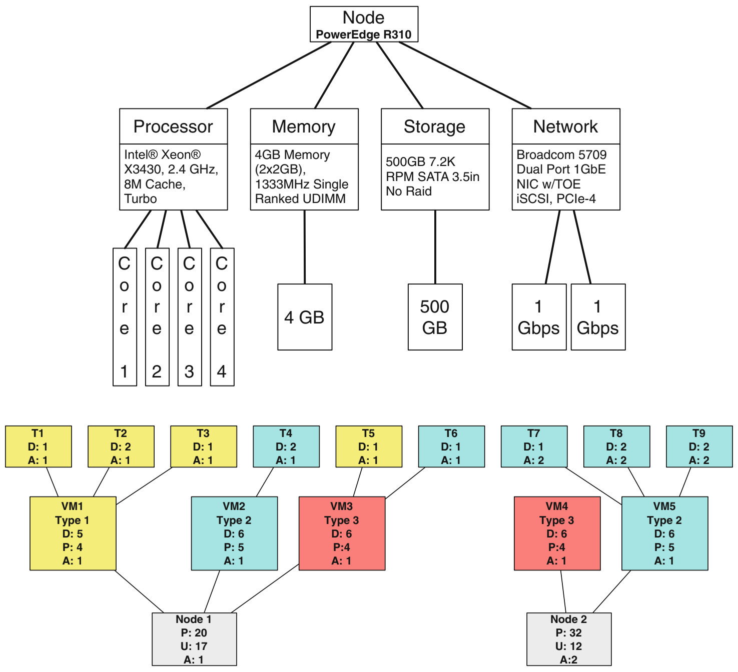


Fig. 1. Example of a representation of a computing node within an HPC cluster

holistic model represents each resource by a resource vector: $res = (typ_1, \dots, typ_z)$. The vector is composed of *resource types* that represents distinct resource types, e.g. CPU, Memory, Disk etc. Each resource type is further expressed as a vector of *resource supplies*: $typ_i = (sup_{i1}, \dots, sup_{iy})$ that represents the exact implementation and number of resource supplies, i.e. hardware components. Finally, each resource supply is defined as $sup_{ij} = (arch_{ij}, cap_{ij})$, where $arch_{ij}$ represents the component architecture (that determines the components characteristics) and cap_{ij} represents the component capacity (e.g. MIPS, RAM or disk size). The architectures can create a partial order, based on the relation of the strict superiority (in terms of performance) of $arch_i$ over $arch_k$, denoted as $arch_k \prec arch_i$. A sample graphical resource vector of a real node is presented on the top of Fig. 1. The node is composed of four resource types. Each of the types is composed of a single supply. In this case these are: CPU with 4 symmetric cores (with capacity expressed in e.g. corresponding number of MIPS), Memory and Storage with single capacity, and Network card wit 2 interfaces. The architecture ordering could be based on the comparison of architectures of this node supplies with other architectures in the same data centre. The resource allocation in a holistic model is represented by resource provisions and resource demands. A *Resource provision* is the representation of the resource offered by a lower layer to the higher layer. Resource provisions are the resources of physical machines and the resources offered by containers to tasks. The *Resource* *demand* is the representation of the resources consumed by higher layer entities and it corresponds to the resources reserved by a container on a machine, or the resources requested by a task from a container. Figure 1 presents also a simple allocation example, where two architecture types ($A = \{1, 2\}$ and $1 \prec 2$) are defined for two nodes, each having various capacity of provisioned resources P. U is the aggregated resource utilization at the machine level, D is the resource demand. The difference between P and D of a container represents its overhead. The colour of VMs and Tasks represents their type: blue tasks can be executed on blue or red VMs, while yellow tasks can be executed on yellow or red VMs. The main issue in such a holistic model is to relate the resource utilization with the obtained performance and power. In this paper, we present a lightweight approach for power modelling. Multiple linear regression is the tool used to accurately predict the impact of the used Machine, Configuration, and Task on the system power. As it is of prime importance to correctly derive the parameters of this model, the best approach requires the collection of concrete observations in a real situation, featuring the virtualization technologies we try to characterize. The next section details the experimental setup performed to reach this goal.

4 Experimental Setup

Two sites of Grid5000, Lyon and Reims were selected for the benchmarking process, as they host two modern HPC clusters: Taurus and StRemi, with diverse hardware architectures and support for Power Distribution Unit (PDU) measurements. An overview of the selected systems we compare in this article is provided in Table 2.

The benchmarking workflow, as presented in Fig. 2, has been described in the following paragraphs. The baseline benchmark uses a customized version of the Grid'5000 squeeze-x64-base image, which contains the benchmark application suite comprised of OpenMPI 1.6.2, GotoBLAS2 1.13, HPCC 1.4.1, Bonnie++ 1.96 and IOzone 3.308. This image is deployed on the target node with the kadeploy3 Grid'5000 utility from the sites' frontend and then the *hypervisor-benchmark-baseline* script is used to launch the benchmark process. This script mounts on the host the site's NFS shared homes, launches in background the dstat utility which is being used to collect resource usage statistics from the node, then starts the *benchmark* script. The *benchmark* script runs HPCC, Bonnie++ and IOzone with cluster-specific values, logging the progress of these applications and archiving their results at the end, along

Table 2. Overview of the two types of computing nodes used in this study.

Platform	Site	Cluster	#cpus/n	#RAM	Processor		R_{peak}
`Intel`	Lyon	Taurus	2	32GB	Intel Xeon E5-2630@2.3GHz	6C	110,4 GFlops
`AMD`	Reims	StRemi	2	48GB	AMD Opteron 6164 HE@1.7GHz	12C	163.2 GFlops

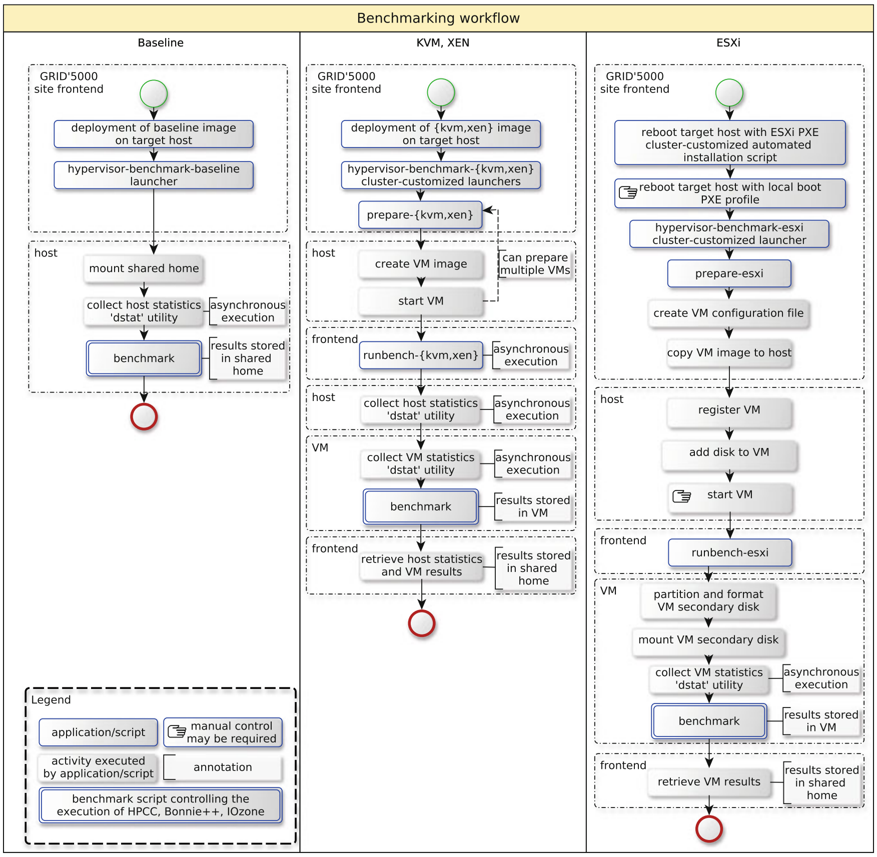

Benchmarking workflow
Baseline
KVM, XEN
ESXi
GRID'5000 site frontend
deployment of baseline image on target host
hypervisor-benchmark-baseline launcher
host
mount shared home
collect host statistics 'dstat' utility
asynchronous execution
benchmark
results stored in shared home
deployment of {kvm,xen} image on target host
hypervisor-benchmark-{kvm,xen} cluster-customized launchers
prepare-{kvm,xen}
create VM image
can prepare multiple VMs
start VM
frontend
runbench-{kvm,xen}
collect host statistics 'dstat' utility
VM
collect VM statistics 'dstat' utility
results stored in VM
retrieve host statistics and VM results
reboot target host with ESXi PXE cluster-customized automated installation script
reboot target host with local boot PXE profile
hypervisor-benchmark-esxi cluster-customized launcher
prepare-esxi
create VM configuration file
copy VM image to host
register VM
add disk to VM
start VM
runbench-esxi
partition and format VM secondary disk
mount VM secondary disk
collect VM statistics 'dstat' utility
retrieve VM results
Legend
application/script
manual control may be required
activity executed by application/script
annotation
benchmark script controlling the execution of HPCC, Bonnie++, IOzone

Fig. 2. Benchmarking workflow.

with the output from the dstat utility. The archive is placed directly in the user's NFS shared home, in a directory which reflects the site name, cluster name, and the job ID of the OAR reservation the user holds. The benchmarking workflow for KVM and XEN is identical, although it is based on different scripts customized to work with these hypervisors. The KVM deployment image has been created from the baseline image, while the XEN image is based on squeeze-x64-xen. Both host images contain VM image files which incorporate the same benchmark suite as the baseline image. After the deployment of the appropriate host image the corresponding *hypervisor-benchmark-*{*kvm,xen*} launcher is started, which contains user-configured parameters that specify how many virtual machines will be configured, the number of virtual cores and memory available to each. The launcher starts the appropriate *prepare-*{*kvm,xen*} script, which in turn connects to the host node, copies and resizes the virtual image (to more than twice the configured VM RAM size, as needed by the Bonnie++ benchmark), pushes the *benchmark* script to it, then starts the VM, pinning the virtual cores to host cores one-to-one. In the next step, a VM-controller *runbench-{kvm,xen}* script is started in the background on the frontend, which waits for the VM to become available on the network, launches the dstat utility in the background on the host and in the VM, then starts the benchmark. When the *benchmark* script has finished, the results archive (containing also the dstat statistics) is retrieved from the VM, along with the host statistics, and the results are placed on the site frontend, in the user's home directory following the same pattern as for the baseline test. The workflow for the ESXi benchmark requires that the target host be booted (through the Grid'5000 kareboot application) with a specific PXE profile and configuration files so that the ESXi installer boots and configures the host according to a cluster-specific kickstart automated installation script. After the installation is done, the host automatically reboots and manual user intervention is required in order to ensure that the host will boot from the local drive by having an *ESXi-install* script reboot the host with another PXE profile that chainloads the newly installed MBR. The ESXi installer has been forced to use a MBR type partitioning scheme, as opposed to its default GPT in order to not interfere with the operation of the Grid'5000 platform. When the ESXi hypervisor has booted, the *hypervisor-benchmark-esxi* user-customized launcher starts the *prepare-esxi* script which then creates an appropriate ESXi VM configuration file and copies it along with the VM image to the host. The script registers the VM and adds a second disk to it as the VM image itself cannot be resized on the host as was the case for KVM and XEN, then starts the VM. This last step may require manual user control, as in some cases the VM is not successfully started automatically. The *runbench-esxi* script is then used to control the VM, in which it partitions, formats and mounts the new disk that was added (which will hold both temporary files from the benchmark and the results), then launches the dstat tool in background and starts the benchmark script. After the benchmark ends, the results archive (with the dstat statistics) is retrieved and stored on the Grid'5000 site's frontend in the same way as for the baseline, KVM and XEN tests. The number of experimental runs for combinations of environments and nodes are presented in Table 3. Due to the need of manual interventions and special preparation of cluster, the ESXi environment was tested only on nodes 7 and 10 in Taurus cluster and nodes 30 and 31 in StRemi cluster. The baseline environment was tested only 4 times on the taurus-8 node due to technical problems with that node that appeared at the end of the sequence of experiments.

Table 3. Number of runs for environment and node.

config:	**baseline**	**KVM**	**Xen**	**VMWare ESXi**	**Observation No.**
stremi-3	5	5	5	0	10916
stremi-6	5	5	5	0	10907
stremi-30	5	5	5	5	13706
stremi-31	5	5	5	5	14026
taurus-7	5	5	5	5	6516
taurus-8	4	5	5	0	4769
taurus-9	5	5	5	0	5085
taurus-10	5	5	5	5	6545

Monitoring of the performance metrics and Data processing. To accurately estimate the status of the analyzed system at a given period of time, a set of performance and power metrics have to be collected. This data was gathered using two monitoring tools: *Ganglia* and *dstat*. Ganglia is a monitoring tool that works at the grid level and is used in this work to gather power readings of the monitored nodes: in the StRemi cluster, instantaneous power readings are available every 3s using SNMP (Raritan) with an accuracy of 7W, while in the Taurus cluster, power is recorded using OmegaWatt power meters that return average power each second with an accuracy of 1W. In both cases, Ganglia aggregates measured values over 15s periods. In order to ensure persistency of values recorded using Ganglia, they are accessed using Grid5000 API and stored in an external database. The utilization metrics of CPU, memory, disk IO and networking IO are recorded using dstat with a frequency of 1s. The recorded metrics and corresponding units are: (1) CPU – user, system, idle, wio (%) (2) Memory – used, buffered, cached, free (B) (3) Disk – read, write (B) (4) Network –received, send (B). Due to the specifics of dstat, missing or duplicated readings are possible. In order to prepare the data for modelling, the dstat values are aggregated (summed for flow values such as IO, averaged for utilization metrics such as *cpu_user*) over 15s periods corresponding to the Ganglia monitoring readings. In case of missing dstat values for periods longer than 15s, the data from Ganglia is discarded from modelling (the amount of data removed in this way is 1.2%, removed observations do not follow any obvious pattern). Such granularity of measurements corresponds to the monitoring utilities used in production systems and diminishes the impact of measuring infrastructure on the system performance. Additionally, each observation has supplemental information about the cluster, node_uid, hypervisor and benchmark phase. The data was preprocessed and statistically analysed using *R* statistical software with the packages *zoo* for data series processing. As a result, 72470 observations were measured, as presented in detail in Table 3.

5 Experimental Results

Performance analysis. While the heart of this study does not reside in the pure performance evaluation of the considered virtualization technology, we present here the average raw performance results obtained over the multiple runs of the HPCC benchmark in each considered configuration. In an attempt to improve the readability of the article, we limit on purpose the number of displayed test results to the ones of HPL, DGEMM, PTRANS and STREAM. First of all, a synthetic view indexed over the hardware architecture is proposed in Fig. 3. We can see that in every single case, the Intel-based configuration outperforms its AMD counterpart, despite a presumed lower peak performance T_{peak}. Then, we illustrate the performances for each test to extract a trend in the relative overhead induced by the considered virtualization technologies. It appears from these plots that the three considered hypervisors offer a similar overhead as regards the computing benchmarks (between 10 to 15%).

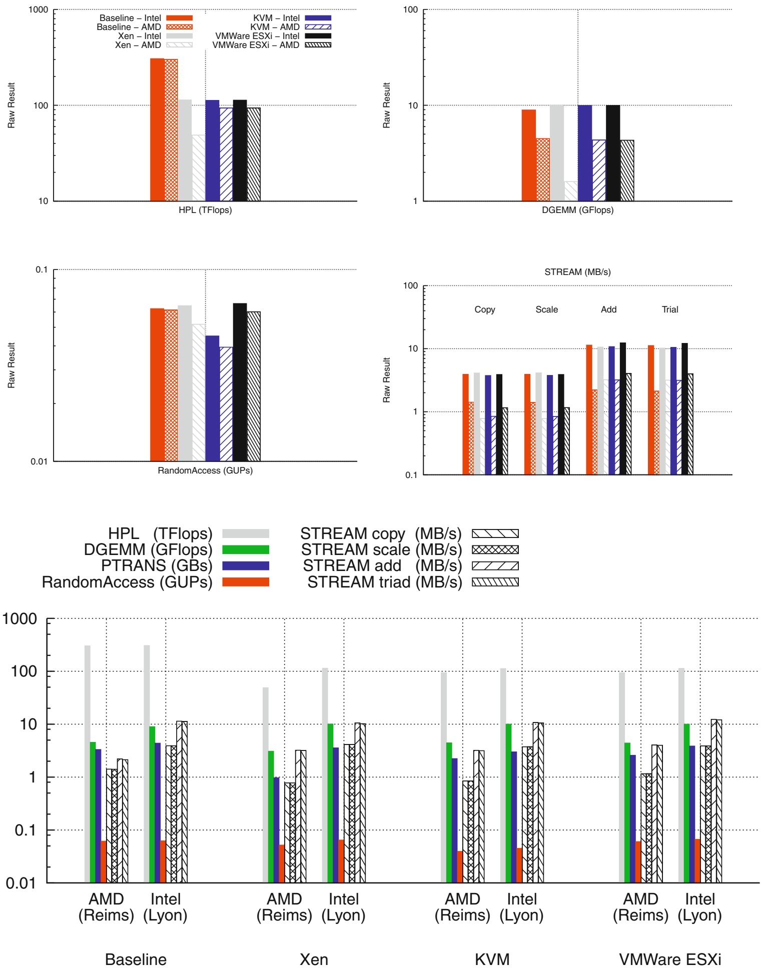


Fig. 3. Average performance of the considered virtualization technologies.

One inherent limit to the usage of virtualization in an HPC environment obviously resides in the huge overhead induced on I/O operations. Thus, we present the results of the IOZone benchmark in Fig. 5. If a significant degradation of the performances is observed as soon as a hypervisor is present, we can see a surprising element as regards the `rewrite` and `random_write` test on the ESXi environment which perform better than the bare-metal system. This is probably due to a better cache strategy on the file system deployed by this environment.

Energy-efficiency analysis and Power modelling by Multiple Regression. For each considered configuration we have measured the energy consumed to run the different benchmarks. As an illustration of the many runs performed, we provide in Fig. 4 traces of selected runs. The analysis of the traces left of each run on the selected configuration permitted to refine the parameters of

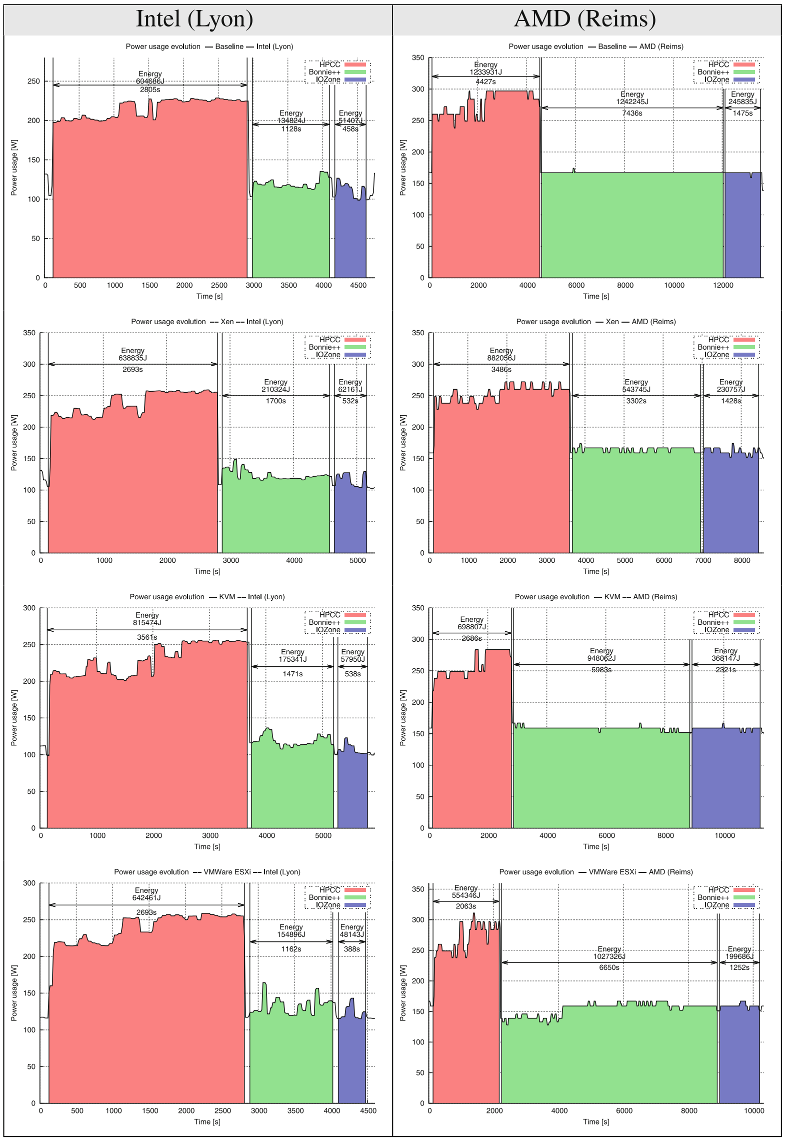


Fig. 4. Power profile of selected runs in each configuration.

the holistic model presented in Sect. 3. We now detail the statistical approach operated in this context.

The presented approach to model power using utilisation metrics as predictors is multiple regression. The advantage of this method is low computational

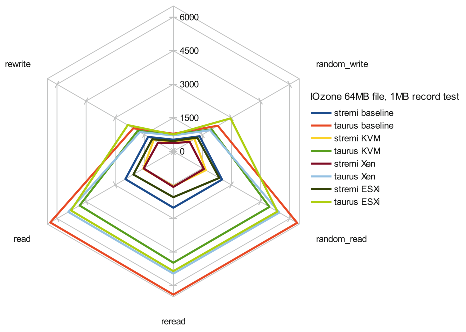


Fig. 5. IOZone results for each configuration.

complexity, no need for parameters and deterministic results. The final model is presented as a linear function of predictors [11]:

$$E(y|x_1, \ldots, x_k) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k, \tag{1}$$

where $E(y|x_1, \ldots, x_k)$ is the expected value of response y given fixed values of regressors $x_1, \ldots, x_k$. The coefficient β_0 is referred to as *intercept* and the other coefficients $(\beta_1, \ldots, \beta_k)$ are called *slopes*. In the context of this work two categories of predictors can be distinguished. *Numerical* predictors are based on the data gathered by monitoring tools. *Categorical* predictors are additional data that group the observations. The aim of modelling at the node level is twofold: to analyse the operation of a system and to predict its behavior. The results of multiple regressions are presented according to concepts of *complete-pooling* and *no-pooling* [12]. In the former case all observations are taken into account disregarding groups specifics, while in the latter case different groups are modelled separately. The following models, varying by sets of predictors, are proposed:

1. *Basic* – all possible predictors taken into account
2. *Refined* – Basic processed by backward stepwise algorithm based on AIC (using default *step* function in R). In all cases it resulted removing only *disk read.*
3. *No Phases* – all possible predictors taken into account but no explicit information about the workload type
4. *CPU Hom.* – only *cpu user* and *cpu idle* taken into account for a simplest homogeneous bottom-line model
5. *CPU Het.* – only *cpu user*, *cpu idle* and *node uid* taken into account for a simplest heterogenous model
6. *No Group* – all possible numerical predictors, no categorical predictors
7. *Cluster-wise* – all possible numerical predictors and cluster predictor, test of homogenous hardware hypothesis (only in complete-pooling setting)
8. *Group Only* – all possible group predictors, no numerical predictors.

Table 4. Complete-pooling models quality

Model	R^2	Residuals						Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%
Basic	0.959	10.4	-116	-3.54	0.8	5.07	117	6.67	3.8%
Refined	0.959	10.4	-116	-3.54	0.806	5.07	117	6.67	3.8%
No Phases	0.941	12.4	-127	-4.13	1.04	4.88	125	8.18	4.4%
CPU Hom.	0.814	22	-147	-13.1	3.9	13.6	160	16.7	9.6%
CPU Het.	0.922	14.2	-129	-5.12	-0.0472	4.89	129	9.73	5.0%
Only phases	0.922	14.3	-114	-3.94	2.06	5.51	72.8	8.75	4.9%
No group	0.856	19.4	-142	-8.45	2.96	11.3	129	14.1	8.0%
Clusterwise	0.928	13.7	-122	-7.22	0.876	7.02	131	9.97	5.3%
Group only	0.924	14.1	-113	-4.29	2.52	5.88	69.9	8.77	4.9%

The results of Complete-pooling analysis are presented in Table 4. The tables in this section are based on three main quality indicators of a model: R^2 value, distribution of residuals including standard error, minimal and maximal values, first and third quartile, and median, and finally the average values of absolute prediction error presented as total error in Watts or relative error in percents. The absolute prediction error was calculated for each observation in the data set as an absolute value of the difference between predicted and observed values of node power. The models that take into account all predictors (*Basic* and *Refined*) have the highest R^2 values, the smallest residual standard error and average prediction error. The *No Phases* model presents similar scores, arguing that knowledge about applications specifics is not necessary if utilisation metrics and environment information are available. The *CPU Het.* model has high R^2 value and acceptable values for the statistics of residuals, contrary to the *CPU Hom.* that is the worst of investigated models. Similarly, the *No Group* model has a low quality, worse than the *Group Only* model that estimates the best value for categorical predictors and has in effect only several possible response values. The comparison of *Cluster-wise* and *CPU Het.* is interesting: the former model has slightly better R^2 values and residual standard error, but the distribution of residuals is better for the latter model, as well as mean absolute prediction errors, pointing out that the nodes in each cluster have heterogenous power consumption, despite their homogenous hardware configuration. Reassuming, the Basic and Refined models are the most accurate and they include all elements of holistic model, confirming the necessity of detailed information for accurate system modelling.

The quality of No-pooling models is presented in Table 5 for clusters StRemi and Taurus. The No-pooling methodology divides the set of data used in the Complete-pooling scenario into two disjoint subset, one for each of the clusters. As a result, the obtained model have distinct slopes of coefficients for two clusters, which is rational considering the differences in the underlying hardware. As a result, No-pooling models have better quality than corresponding Complete-pooling models. The quality of No-pooling modelling is apparently worse than the complete-pooling model for the Taurus cluster, but it must be taken into account that because of the longer running time of experiments on StRemi nodes, there are more observations from this cluster, that bias the Complete-pooling

Table 5. No-pooling models quality

No-pooling StRemi cluster models quality

Model	R^2	Residuals						Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%
Basic	0.968	8.74	-105	-2.9	0.398	3.52	106	5.28	2.8%
Refined	0.968	8.74	-105	-2.9	0.396	3.52	106	5.28	2.8%
No Phases	0.955	10.4	-114	-3.24	0.785	3.51	121	6.15	3.1%
CPU Hom.	0.925	13.3	-116	-7.07	0.0893	7.98	128	9.44	4.7%
CPU Het.	0.938	12.2	-120	-4.2	1.04	4	125	7.8	3.7%
Only phases	0.956	10.2	-104	-3.69	1.3	4.31	112	6.3	3.2%
No group	0.942	11.7	-120	-6.33	1.05	6.78	126	8.19	4.3%
Group only	0.96	9.77	-104	-3.07	0.694	4.11	112	5.87	3.1%

No-pooling Taurus cluster models quality

Model	R^2	Residuals						Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%
Basic	0.957	11.6	-117	-6.16	0.0846	5.97	132	7.62	4.7%
Refined	0.957	11.6	-117	-6.16	0.08	5.97	132	7.62	4.7%
No Phases	0.925	15.2	-127	-9.04	0.967	10.8	142	11.2	6.3%
CPU Hom.	0.896	17.9	-128	-10.6	-1.49	12.2	137	13.5	7.4%
CPU Het.	0.898	17.7	-131	-10.7	-2.18	12.3	133	13.4	7.4%
Only phases	0.884	18.9	-114	-6.75	1.33	9.28	57.4	11.9	7.3%
No group	0.918	15.9	-127	-9.05	0.0862	11.9	141	11.8	6.6%
Group only	0.902	17.4	-112	-5.95	0.883	7.59	56.4	10.2	6.6%

model and its accuracy towards this cluster results. The more interesting fact is the worse prediction results for the Taurus cluster, which has more accurate power measurement infrastructure. This result confirms that power modeling of a node can be done on longer time intervals without increasing the The sample results of using power prediction by the Refined non-pooling models, identified as the best ones, are presented in Fig. 6. The figure presents the predicted by Refined model power consumption against the observed values for sample runs for each combination of hardware and hypervisor. The less accurate prediction of ESXi can be explained by distinct hypervisor engine or limited amount of samples for this hypervisor. Despite that, the model is able to accurately follow the power consumption pattern for each configuration.

The results of models can be used to create an instance of holistic model. In such case, given the used node, hypervisor, and utilisation levels of hardware components of a selected machine, one can predict the final power output. The model can be further refined by observation of the phases of computation. General utilisation of holistic model for decision making is presented in this section. Table 6 presents the difference between nodes in the two clusters as well as between nodes with the same hardware configuration. In this case, the difference between clusters is approximately 40W. The difference between nodes in StRemi cluster is up to 14W, while among Taurus nodes it is up to 6W. The phases have large impact on the final power consumption in derived model, as presented in Table 6. The presented values are *adjustments* to the amount based on the utilization metrics. Therefore, there is no sense in straight comparison of these values (e.g. nodes do not consume approximately 3W *more* in idle state than during the STREAM phase). However, these values present that knowledge about running application characteristics can add valuable information to

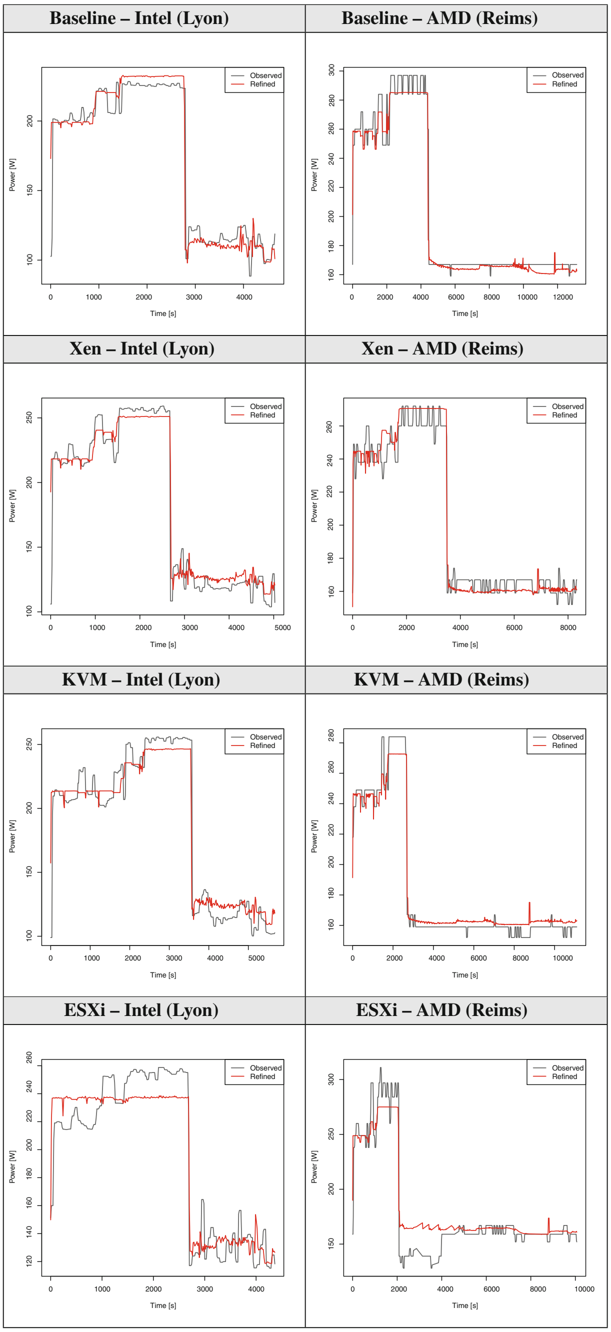


Fig. 6. The predicted energy profiles for the best identified models.

Table 6. The Complete-pooling model coefficients for nodes, phases and hypervisors, together with the model numerical coefficients.

stremi-3	stremi-30	stremi-31	stremi-6	taurus-10	taurus-7	taurus-8	taurus-9
0	-1.8	-14	-4.7	-45	-40	-46	-44

Bonnie	DGEMM	FFT	HPL	IOZONE	PTRANS	RandomAccess	STREAM	idle
0	5.7	6.5	16	0.012	-6.1	-11	3.1	6.1

ESXi	KVM	Xen	baseline
0	-3.6	-5.4	-19

Intercept	cpu user	cpu system	cpu idle	cpu wio	mem used
316	-0.78	-1.3	-1.7	-1.8	1.1E-9

mem buffers	mem cached	mem free	disk write	bytes rec.	bytes send
-3.1E-08	8.0E-10	8.3E-10	-1.8E-10	4.6E-05	-1.1E-04

power predictions. The hypervisor type influence is also depicted. The presented values show a gap between hypervisors and baseline. However, it is important to remember that performance metrics were collected at the container (guest VM) level, therefore these differences may be adjustments for the resources consumed by the hypervisor, which were not used for modeling. Finally, we discuss the numerical predictors. The intercept has high positive value. The CPU utilization coefficients are negative. The most power consuming mode is cpu user, followed by cpu system. Cpu idle and cpu wio are the least power-consuming modes of operation. The memory in used, cached, and free states has significantly higher power consumption that in buffered state. The output activities generally decrease the power of the system, which is coherent with the cpu wio values and may be caused by entering cpu into lower power states during large output operations. Contrary to that, network receive state increases the system power, however less significantly than the decrease of network send.

6 Conclusion

In the paper we introduce and experimentally evaluate a holistic model based on its power estimation. The holistic power modelling is able to represent power consumption with an average relative error of 3.8%. The model is lightweight, as it is represented by a single equation, and its creation also has a low time complexity. It can be extended after creation by adding or modifying the coefficients, thus it can adapt to dynamic systems. The model could be further refined using Partial-pooling techniques with varying slopes, to benefit from an intermediary approach between Complete-pooling and No-pooling [12]. The parameters of this model have been refined by a set of concrete experiments on the Grid'5000 platform that permitted the careful analysis and modelling of three widespread virtualization, namely Xen, KVM and VMware ESXi on the two leading hardware architectures (AMD and Intel). When compared to a *baseline* environment running in native mode, we have seen that the usage of hypervisors in an HPC environment raises a limited overhead, and can be foreseen as soon as the I/O operations are correctly handled, as the computing performances are nearly identical between the considered virtualization technologies. In this sense, we confirm the results of preceding studies. The future work includes adding more elements to holistic models, such as environmental metrics (temperature, supplied voltage), examining the effect of multi tenancy and overhead of cloud management systems (e.g. OpenNebula, OpenStack), modelling the performance of configurations, considering network-intensive load and parallel tasks, and building such models based on the hardware component of node (directly using the resource vector concept to build the power model), rather than full hardware platform. All that work require further experimentation on a larger set of applications and machines.

Acknowledgments. The experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr). This work was completed with the support of the INTER/CNRS/11/03/ Green@cloud Project and the Cost action IC0804.) M. Guzek acknowledges the support of the FNR in Luxembourg and Tri-ICT, with the AFR contract no. 1315254. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804. The authors would like also to thank Hyacinthe Cartiaux and the technical committee of the Grid'5000 project for their help and advises in the difficult deployment of the ESXi environment.

References

1. Bonnie++. [online] http://www.coker.com.au/bonnie++/
2. Grid'5000. [online] http://grid5000.fr
3. Iozone filesystem benchmark. [online] http://www.iozone.org/
4. A. Gavrilovska et al.: High-performance hypervisor architectures: virtualization in HPC systems. In: Proceedings of HPCVirt 2007, Portugal, March 2007
5. Ali, Q., Kiriansky, V., Simons, J., Zaroo, P.: Performance evaluation of HPC benchmarks on VMware's ESXi server. In: Alexander, M., et al. (eds.) Euro-Par 2011, Part I. LNCS, vol. 7155, pp. 213–222. Springer, Heidelberg (2012)
6. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the Art of Virtualization. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles, SOSP '03. pp. 164–177. ACM, New York (2003)
7. Bohra, A., Chaudhary, V.: Vmeter: power modelling for virtualized clouds. In: 2010 IEEE International Symposium on Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), pp. 1–8 (2010)

8. Chen, Q., Grosso, P., van der Veldt, K., de Laat, C., Hofman, R., Bal, H.: Profiling Energy Consumption of vms for Green Cloud Computing. In: Proceedings of the 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, DASC '11, pp. 768–775. IEEE Computer Society, Washington, DC (2011)
9. Economou, D., Rivoire, S., Kozyrakis, C.: Full-system power analysis and modeling for server environments. In: Workshop on Modeling Benchmarking and Simulation (MOBS) (2006)
10. Fa, X., Webe, W.-D., Barroso, L.A.: Power provisioning for a warehouse-sized computer. In: Proceedings of the 34th Annual International Symposium on Computer Architecture, ISCA '07, pp. 13–23. ACM, New York (2007)
11. Fox, J., Weisberg, S.: An R Companion to Applied Regression. SAGE Publications, Los Angeles (2011)
12. Gelman, A., Hill, J.: Data Analysis Using Regression and Multilevel/Hierarchical Models. Cambridge University Press, Cambridge (2009)
13. Dongarra, P.L.J.J.: Introduction to the hpcchallenge benchmark suite. Technical report, ICL (2004)
14. Kansal, A., Zhao, F., Liu, J., Kothari, N., Bhattacharya, A.A.: Virtual machine power metering and provisioning. In: Proceedings of the 1st ACM Symposium on Cloud Computing SoCC '10, pp. 39–50. ACM, New York (2010)
15. Kivity, A., et al.: kvm: the Linux virtual machine monitor. In: Ottawa Linux Symposium, pp. 225–230, July 2007
16. Liu, L., Wang, H., Liu, X., Jin, X., He, W.B., Wang, Q.B., Chen, Y.: Greencloud: a new architecture for green data center. In: Proceedings of the 6th International Conference Industry Session on Autonomic Computing and Communications Industry Session, ICAC-INDST '09, pp. 29–38. ACM, New York (2009)
17. Younge, A.J., Henschel, R., Brown, J., von Laszewski, G., Qiu, J., Fox, G.C.: Analysis of virtualization technologies for high performance computing environments. In: The 4th International Conference on Cloud Computing (IEEE CLOUD: Washington, DC), 07/2011. IEEE (2011)

Runtime Scheduling of the LU Factorization: Performance and Energy

Pedro Alonso[1](✉), Manuel F. Dolz[2], Francisco D. Igual[3], Enrique S. Quintana-Ortí[2], and Rafael Mayo[2]

[1] Depto. de Sistemas Informáticos y Computación, Universitat Politècnica de València, 46022 Valencia, Spain
palonso@dsic.upv.es

[2] Depto. de Ingeniería y Ciencia de Computadores, Universitat Jaume I, 12071 Castellón, Spain
{dolzm, quintana, mayo}@uji.es

[3] Depto. de Arquitectura de Computadores y Automática, Universidad Complutense de Madrid, 28040 Madrid, Spain
figual@fdi.ucm.es

Abstract. In this paper, we enhance the SuperMatrix runtime scheduler from the `libflame` library for dense linear algebra in two different directions that address high performance and energy. First, we extend the runtime scheduler by accommodating hybrid CPU-GPU executions and managing task priorities for dense linear algebra operations, with remarkable performance improvements. Second, we introduce techniques to reduce energy consumption during idle times inherent to parallel executions, attaining fair energy savings. While our techniques are applicable to the complete `libflame` library, in this paper we use the LU factorization with partial pivoting to illustrate the actual impact on performance and energy consumption of the adopted techniques.

Keywords: Energy-aware computing · High performance · Hybrid architectures · Runtime task schedulers · Dense linear algebra

1 Introduction

The *power wall* is a major hurdle that the scientific community will need to tackle in order to build the Exascale systems that are expected to be deployed by the end of this decade [7,10]. In the last years, continued pressure to further reduce power consumption signals a trend towards heterogeneous designs that combine hardware accelerators like GPUs (graphics processors) or DSPs (digital signal processors), with general-purpose multicore technology [7,10]. GPUs in particular have arisen as an appealing architecture for compute-intensive, data-parallel applications, mainly due to their vast amount of hardware concurrency, affordable price, favorable energy-performance ratio, and the existence of *de facto* programming standards such as CUDA and OpenCL. Nevertheless, programming a hybrid platform consisting of one to several multicore processors

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 153–167, 2013.
DOI: 10.1007/978-3-642-40517-4_14,

and multiple GPUs is still a considerable challenge if a significant fraction of the machine peak performance is to be attained.

Following a path pioneered by Cilk [8], a number of *runtime schedulers* have been proposed in recent years to address the increase of hardware concurrency and, in some cases, the heterogeneity of recent architectures. OmpSs [12] and StarPU [17], among others, propose implicit parallel programming models with dependence analysis, particularly adapted to exploit task-level parallelism and alleviating the programmability problem. For the specific domain of dense linear algebra (DLA), recent studies with SMPSs (a precursor of OmpSs) [4], StarPU, Quark [14] and SuperMatrix [15] have demonstrated the advantage of extracting task-level parallelism from DLA operations using this same approach.

SuperMatrix is a runtime that was specifically designed from its inception for the execution of DLA operations, enabling a seamless execution of the full functionality of the `libflame` DLA library [18] on a variety of parallel platforms, from multicore desktop servers [15] and hybrid systems equipped with CPU-GPU [16] to small-scale clusters [11].

In this paper, we describe some major changes to SuperMatrix that significantly improve its performance on hybrid CPU-GPU systems and introduce energy-aware policies in the operation of the runtime. In particular, we make the following contributions:

- The initial version of the SuperMatrix runtime committed one CPU thread per GPU that was responsible, among other work, of scheduling ready tasks for execution in the attached accelerator. This control thread ran in one CPU core, and no provision was made to exploit additional CPU cores in case they outnumbered the GPUs. In the new runtime scheduler we accommodate one thread per CPU core in the system. Among these, there is one control thread per GPU, but now there is also one worker thread for each one of the remaining CPU cores. Therefore, in the new runtime scheduler we can leverage any combination of hardware CPU-GPU concurrency in the platform to increase performance.
- Additionally, we introduce priorities in the runtime to advance the computation of tasks that lie on the critical path of the algorithm. As our experiments will illustrate, the outcome is a significant reduction of idle time, especially for hybrid CPU-GPU platforms, and faster execution.
- We integrate two energy-aware techniques into SuperMatrix, extending the results in [3] to the new runtime scheduler.
- Finally, we analyze the practical impact of the performance enhancements and energy-saving strategies using the LU factorization with partial pivoting, a key dense linear algebra operation representative of several other matrix factorizations for the solution of dense linear systems.

```
Algorithm: $[A] := \text{LU\_BLK}(A)$
Partition $A \rightarrow \left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right)$
    where $A_{TL}$ is $0 \times 0$
while $n(A_{TL}) < n(A)$ do
  Determine block size $b$
  Repartition
    $\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \rightarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right)$
      where $A_{11}$ is $b \times b$

    % Factorize current panel
    $\left[\left(\begin{array}{c} A_{11} \\ \hline A_{21} \end{array}\right)\right] := \text{LU\_UNB}\left(\begin{array}{c} A_{11} \\ \hline A_{21} \end{array}\right)$
    % Update trailing submatrix
    $A_{12} := \text{TRSM}(A_{11}, A_{12})$
    $A_{22} := \text{GEMM}(A_{21}, A_{12}, A_{22})$

  Continue with
    $\left(\begin{array}{c|c} A_{TL} & A_{TR} \\ \hline A_{BL} & A_{BR} \end{array}\right) \leftarrow \left(\begin{array}{c|c|c} A_{00} & A_{01} & A_{02} \\ \hline A_{10} & A_{11} & A_{12} \\ \hline A_{20} & A_{21} & A_{22} \end{array}\right)$
endwhile
```

```
Algorithm: $A := \text{TRSM}(L, A)$
 Partition $A \rightarrow (A_L \,|\, A_R)$
    where $A_L$ has 0 columns
while $n(A_{TL}) < n(A)$ do
  Determine block size $b$
  Repartition
    $(A_L \,|\, A_R) \rightarrow (A_0 \,|\, A_1 \,|\, A_2)$
      where $A_1$ has $b$ columns

    % Solve for current panel
    $A_1 := L_1^{-1} A_1$

  Continue with
    $(A_L \,|\, A_R) \leftarrow (A_0 \,|\, A_1 \,|\, A_2)$
endwhile
```

```
Algorithm: $A := \text{GEMM}(A, B, C)$
 Partition $B \rightarrow (B_L \,|\, B_R)$ , $C \rightarrow (C_L \,|\, C_R)$
    where $B_L$ has 0 columns, $C_L$ has 0 columns
while $n(B_{TL}) < n(B)$ do
  Determine block size $b$
  Repartition
    $(B_L \,|\, B_R) \rightarrow (B_0 \,|\, B_1 \,|\, B_2)$ ,
    $(C_L \,|\, C_R) \rightarrow (C_0 \,|\, C_1 \,|\, C_2)$
      where $B_1$ has $b$ columns, $C_1$ has $b$ columns

    % Update current panel
    $C_1 := C_1 - A \cdot B_1$

  Continue with
    $(B_L \,|\, B_R) \leftarrow (B_0 \,|\, B_1 \,|\, B_2)$ ,
    $(C_L \,|\, C_R) \leftarrow (C_0 \,|\, C_1 \,|\, C_2)$
endwhile
```

Fig. 1. Blocked algorithm for the LU factorization (left) and procedures for the update of the trailing submatrix (top and bottom right). In the algorithms, LU_UNB$(\cdot)$ is a procedure that computes the LU factorization of a panel using an unblocked algorithm, and $n(\cdot)$ is the operator that returns the number of columns of its argument.

2 SuperMatrix Data-Flow Parallel Runtime for DLA

2.1 Principles of Data-Flow Execution

The first step to obtain a task parallel data-flow execution of a DLA operation is to derive a (sequential) blocked algorithm, decomposing the computation that needs to be performed into suboperations, or *tasks*, of a certain granularity. Consider, e.g., the LU factorization of a (nonsingular) matrix $A \in \mathbb{R}^{n \times n}$, which computes the decomposition $A = LU$, where $L \in \mathbb{R}^{n \times n}$ is unit lower triangular and $U \in \mathbb{R}^{n \times n}$ is upper triangular. (For simplicity, we neglect pivoting during the following presentation, though in practice, as well as in all our experiments, it is included for numerical stability.) Figure 1 left presents a right-looking blocked algorithm for this factorization using the FLAME notation [6].

Armed with the blocked algorithm, the next step consists in identifying the dependencies among the suboperations/tasks. For this purpose, it suffices to determine the order in which tasks appear in the algorithm as well as the operands

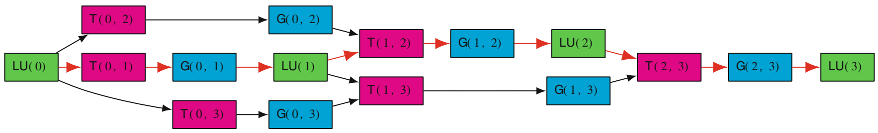


Fig. 2. TDG for the LU factorization with partial pivoting of a matrix A consisting of $s \times s = 4 \times 4$ blocks. Red arrows identify the critical path of the algorithm.

that each task reads (inputs), writes (outputs), or reads/writes (inputs/outputs). For the particular case of an $n \times n$ matrix A consisting of $s \times s = 4 \times 4$ blocks of dimension $b \times b$ each (i.e., $n = s \cdot b$), this analysis yields the task dependency graph (TDG) in Fig. 2. There, LU(k) stands for the factorization of the k-th panel (column block), while $\mathsf{T}(k, j)$ and $\mathsf{G}(k, j)$, refer, respectively, to the triangular system solve (Trsm) and the matrix-matrix update (Gemm) of the j-th panel w.r.t. the factorization of panel k (see Fig. 1 (right)).

In summary, the TDG associated with a given operation/algorithm dictates different "correct orderings" in which the tasks can be computed. Current runtimes leverage this information to produce an out-of-order, data-flow schedule of the TDG and a concurrent execution of the tasks.

2.2 Details on SuperMatrix

In the SuperMatrix runtime, the dependence analysis is performed by a single thread, before the actual computation commences. This thread inspects the code, inserting tasks (i.e., invocations to perform suboperations) as they are encountered into a data structure that captures the information of the TDG in the form of a *work queue*. The only exception to this is the root task of the graph which, having no dependencies with other suboperations, is inserted directly into the *ready queue*. We have experimentally determined that, given the dimension of the DLA problems that are usually tackled using hybrid architectures, the cost (in time) and memory requirements of the analysis stage are negligible.

After this initial analysis stage, the runtime task scheduler is ready to execute the computations represented by the TDG. For that purpose, a collection of worker threads poll the ready queue for new work. Upon dequeuing a task from this structure, a thread executes the corresponding computation and, once completed, checks which dependencies have been fulfilled, moving those tasks with all dependencies satisfied to the ready queue. Our experiments show that, in general, the overhead introduced by the operation of runtime (i.e., cost of checking dependencies, moving tasks between data structures, etc.) is negligible compared with the volume of arithmetic computation intrinsic to DLA operations.

The original version of SuperMatrix, introduced in [16], employs one control thread per GPU (device) of the target platform. These threads run on a (CPU) core of the host, and are in charge of *i)* monitoring and updating the dependence queues; *ii)* guiding the associated accelerator by carrying out the necessary data transfers and dispatching tasks for execution there; and *iii)* executing computational work that is not suited to the GPU. For example, in our implementation of the LU factorization with partial pivoting, the panel factorizations are performed by the CPU cores as the calculation of the pivot elements in this type of operations requires a fine control that renders them inappropriate for the data-parallel architecture of the GPU. The triangular solves (Trsm) and matrix-matrix updates (Gemm) of the remaining blocks, on the other hand, are performed in the GPUs.In that initial version, no attempt was made to exploit the existence of more CPU cores than GPUs in the target platform.

For the particular case of platforms with multiple memory address spaces, SuperMatrix considers GPU memories as full-associative caches and applies policies such as LRU, write-invalidate and write-update to maintain the coherence of the information stored in the main memory and the GPU memories [16]. These techniques yield a significant reduction of the volume of communications between CPUs and GPUs, diminishing the negative impact of the slow PCI-e bus. Handling with a cache-memory coherence protocol in software, instead of hardware, could be presumed to be inefficient. However, we are dealing here with compute-intensive tasks that perform $\mathcal{O}(b^3)$ arithmetic floating-point operations (flops), with $b \geq 64$ in general, and the cost of maintaining the coherence mechanism was found to be minor.

3 Environment Setup

All the experiments reported in the next two sections were obtained using IEEE double-precision arithmetic on a server equipped with two Intel Xeon E5440 processors (4 cores per socket) at 2.83 GHz and 16 Gbytes of RAM, connected to an NVIDIA Tesla S2050 (4 "Fermi" GPUs). We will refer to this platform as TSL in the rest of the paper. Highly tuned implementations of BLAS and LAPACK were provided by MKL 10.0.1, and the SuperMatrix runtime in `libflame` release 5.0–r6719 were employed in the evaluation. The codes for the LU factorization were those from `libflame`. In all cases, we conducted a detailed experimental analysis in order to identify the optimal block size b for each problem dimension. Unless otherwise stated, results are reported only for this optimal value.

Energy consumption was measured using an APC 8653 PDU (Power Distribution Unit) which samples power at 1 Hz. This device was directly attached to the cable that connects the electric socket to the computer power supply unit. A daemon application ran on a separate tracing server, collecting power samples from the PDU. The measurement application was built on top of our library `pmlib` [1], which is designed to interact with power measurement devices.

4 Improving SuperMatrix Performance

4.1 Performance Analysis of the Original SuperMatrix

The original SuperMatrix scheduler supported two different basic configuration modes, depending on the hardware resources of the target platform:

- *Multicore mode.* One worker thread is bound to each CPU core, executing tasks on it by invoking optimized (sequential) BLAS/LAPACK kernels.
- *MultiGPU mode.* The scheduler performs the execution of tasks on platforms equipped with multiple hardware accelerators -specifically GPUs- transparently to the user. One control thread running on a CPU core is attached during the complete parallel execution to each GPU; and tasks are executed on the GPU using a specific implementation of BLAS for these devices — concretely, CUBLAS for NVIDIA GPUs. The CPU threads are in charge of performing the necessary data transfers between memory spaces prior to any task execution. In case a task is not appropriate for the GPU, the computation can be carried out in the associated CPU core.

The top-left plot in Fig. 3 shows the performance of the LU factorization with partial pivoting, in terms of GFLOPS (1 GFLOPS = 10^9 flops/sec.), using 8 CPU cores (multicore mode) and 4 GPUs (multiGPU mode) of TSL.

From this initial experiment, it is possible to extract a few conclusions, for three different cases according to the matrix dimension. For matrices of size smaller than 4,500, the multicore mode outperforms the performance of the

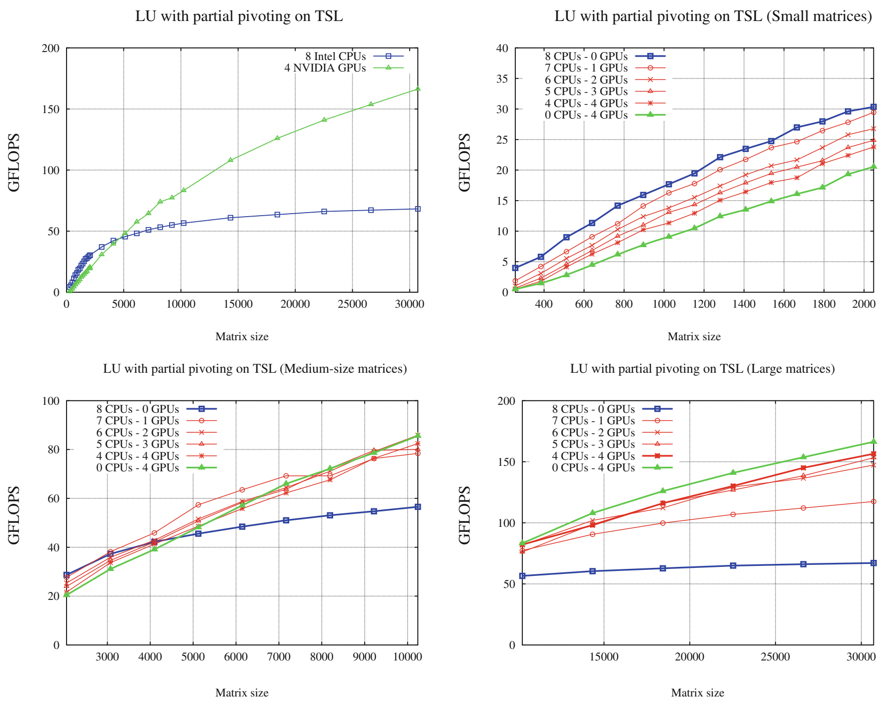


Fig. 3. Performance of the LU factorization with partial pivoting on TSL, using 8 CPU cores (multicore mode) and 4 GPUs (multiGPU mode) (top-left) and a variety of combinations for small problems (top-right), medium-size problems (bottom-left), and large problems (bottom-right).

GPU-based alternative. This is a known result [5,16] as GPUs need large volumes of computation in order to hide memory latencies and data transfer overheads. For matrices in the range of 4,500–5,500, both the multicore and multiGPU modes deliver similar GFLOPS rates. From our observations, this behaviour is common to other dense linear algebra operations. For matrices of size starting at 5,500, the multiGPU setup clearly outperforms the multicore counterpart, basically due to the much higher hardware concurrency of the GPUs.

4.2 Tuning the Scheduler

In past work, several heuristics have been applied to tune the performance of runtime task schedulers by reducing idle time and/or minimizing data movement. In our extension of the SuperMatrix runtime for hybrid CPU-GPU architectures, decisions taken at runtime can address performance, but also energy efficiency:

- The GFLOPS analysis in terms of the problem size reported in the previous subsection determines an optimal mode for each problem size. To leverage this, the runtime system could modify, at execution time, the number of each type of computational resources (CPU or GPU) that are devoted to the actual task computations.
- For the range of matrices labelled as *medium-size problems* identified in the previous subsection, the attained performance is similar for two or more combinations (modes) of the number of CPUs/GPUs. The question thus becomes which mode is more efficient from the point of view of energy consumption.

The next two subsections address the performance goal while the discussion related to energy is delayed till Sect. 5.

4.3 Leveraging Full Hardware Concurrency

In the original SuperMatrix implementation, the execution of tasks was performed either by the CPU cores (multicore mode) or by the GPUs (multiGPU mode). As an exception, in the multiGPU mode a few types of tasks could be executed on the CPU cores, due to their characteristics. While this occurred, though, the corresponding GPUs remained stalled, waiting for the completion of the task. Thus, for example, if four GPUs were used from a platform equipped with 12 CPU cores in the multiGPU mode, only the four cores assigned to guide the GPUs were effectively utilized for GPU management and task dispatching, while the remaining eight CPU cores were wasted.

An improvement to this original execution model considers the GPUs and all the CPU cores as potential workers. In this case, each task type is bound to two different kernel instances, one for the GPU and one for the CPU. Depending on the type of thread a task is mapped to, the corresponding kernel instance is invoked. Data transfers are handled transparently by the runtime, depending on the type of worker thread and the location of the necessary data when the task is dispatched for execution.

The top-right and two bottom plots in Fig. 3 report the performance of the LU factorization with partial pivoting using the modified scheduler that accommodates hybrid executions. In the plots, red lines identify hybrid configurations, while blue and green lines identify multicore and multiGPU configurations, respectively. We illustrate the results by dividing the overall performance lines into three different cases, depending on the size of the involved problem.

– *Small problems (top-right in Fig.* 3). The addition of GPUs to the basic multicore setup does not improve performance for these particular problem sizes. Hybrid executions perform better than the multiGPU counterpart, increasing performance as the number of GPUs decreases. From these results, it is clear that the progressive activation of GPUs for small problems degrades performance, and only the CPU cores should be used in this case.
– *Medium-size problems (bottom-left in Fig.* 3). The insights involve performance and energy-related considerations. Regarding *performance*, hybrid configurations perform better than multicore and multiGPU configurations for most problems sizes in the range. This fact gives the hybrid scheduler different options to select the most appropriate execution configuration at runtime to tune performance for a given problem size. Regarding *energy consumption*, note that there appear problem sizes for which the performance lines for different configurations intersect each other, identifying cases which offer equivalent efficiency. For example, consider the performance attained by the multicore configuration and a hybrid setup using 7 CPU cores/1 GPU for $n = 3,072$. While the performance attained is barely identical, the efficiency of the execution for both configurations is likely to vary significantly when employing or not the GPU. Under this type of situations, the hybrid scheduler can take decisions based on energy-aware considerations.
– *Large problems. (bottom-right in Fig.* 3). In this case, the situation is the opposite than that observed for small problems: the addition of GPUs to the basic multicore setup clearly improves performance, with the multiGPU setup being the most convenient for this particular problem sizes.

4.4 Advancing Critical Tasks

Let us analyze next the task schedule of the LU factorization for a matrix, e.g., of size n = 10,240, with block size b = 1,024, using the multiGPU mode on 4 GPUs. Given these dimensions, the algorithm in Fig. 1 partitions the matrix into s = 10,240/1,024 = 10 panels. At each iteration $k = 1, 2, \dots, s$, the algorithm proceeds by initially decomposing the k-th panel of the input matrix (LU(k)); and next updating the trailing submatrix panelwise w.r.t. the factorization of this panel, which is performed as a sequence of triangular system solves and matrix-matrix udpates (tasks TRSM(k, j) and GEMM(k, j), respectively, with $j = k + 1, k + 2, \dots, s$). For simplicity, hereafter we will refer to the combined application of TRSM and GEMM to the j-th panel as UPDATE(k, j).

Figure 4 (top) shows a trace of the execution of this LU factorization governed with the original SuperMatrix scheduler, with tracing capabilities provided by

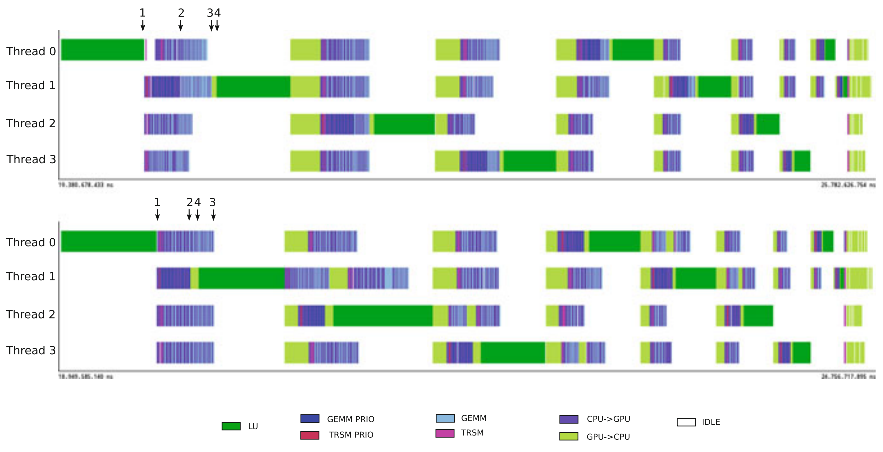


Fig. 4. Traces of the execution of the LU factorization with partial pivoting of a matrix of dimension n=10,240, with b=1,024 using 4 GPUs of TSL, without priority tasks (top) and with priority tasks (bottom). Selected execution points: 1: End of LU(0); 2: End of UPDATE$(0,1)$; 3: End of UPDATE$(0,s)$; 4: Start of LU(1).

`Extrae` and `Paraver` [13]. Note how, in the operation of the original runtime, the factorization LU$(k+1)$ does not commence till the update of the full trailing submatrix w.r.t. the factorization of the previous panel has been completed. An inspection of the order in which tasks are executed there (see the instants marked with numbers 1 to 4 in the trace) reveals that the task LU(1) (execution point 4) does not proceed until the complete update of the trailing submatrix UPDATE$(0,s)$ (execution point 3), is completed.

However, the factorization of the current panel and the update of the first panel of the trailing submatrix both lie on the critical path of the TDG (see Fig. 2) and, therefore, their execution should proceed as soon as possible. Indeed, the task LU$(k+1)$ could effectively start as soon as the task UPDATE$(k,k+1)$ is completed, since the dependencies determine that it is unnecessary to wait for the update of all remaining panels in the trailing submatrix: UPDATE$(k,k+2)$,...,UPDATE(k,s). Thus, the goal of our optimization is to enforce a fast execution of those tasks in the critical path, that in practice yields an overlapped execution of LU$(k+1)$ with UPDATE$(k,k+2)$,...,UPDATE(k,s).

To accomplish this, tasks in the critical path receive a special treatment in the enhanced version of the SuperMatrix scheduler. Specifically, *i)* critical (or priority) tasks are executed as soon as possible to avoid unnecessary stalls; and *ii)* they are mapped to the fastest computational resource (CPU or GPU) available. Let us further elaborate on this.

In principle, the original implementation of SuperMatrix employs a single ready queue containing all tasks with their data dependencies satisfied. (Indeed, when data affinity was in place, there was one queue of ready tasks per thread, containing ready tasks to be run by that thread). In the version enhanced with priorities, the scheduler has been modified to introduce an additional *priority* *queue* (or one priority queue per thread in case data affinity is used), which is managed in the following manner:

– After the execution of a task and the necessary updates of the data dependencies, tasks that become ready are removed from the work queue. If the ready task is critical, it is inserted in the thread's *priority queue*; otherwise, it is moved to a *non-priority queue* shared by all threads.
– When a worker thread becomes idle, it polls the queues of ready tasks in order to obtain a new candidate for execution. Consider first the multiGPU mode. If the thread is in control of a GPU, it first checks the corresponding priority queue; if no priority task is available, the shared non-priority queue is polled for a new task. On the other hand, if the thread runs on a CPU core with no GPU attached, only the non-priority queue is polled. In the multicore mode, the priority queue is always checked before the non-priority one. This forces the runtime to execute critical tasks as soon as they become ready, doing it on the GPU in the multiGPU mode to ensure a fast execution.

The effect of the enhanced runtime is illustrated in the bottom trace of Fig. 4: If we inspect the order in which tasks are executed there, we see that LU(1) (execution point 4) now commences as soon as the update of the first panel of the trailing submatrix has been completed UPDATE(0, 1) (execution point 3), effectively overlapping the execution of tasks from both iterations. In theory, this reduces the idle time, activating new ready tasks that depend on those in the critical path, and accelerating parallel execution.

The performance impact of the improvements introduced in the new runtime are reported in Fig. 5. There, we compare the efficiency of the original version of SuperMatrix with that of the new runtime using the multicore mode and the multiGPU mode of the runtime scheduler. The results show that, especially for large matrices, the performance improvement is remarkable. For the multicore mode, the acceleration varies between 10% and 12%, while for the multiGPU mode, the speed-ups range from 20% to 25% for the largest tested matrices.

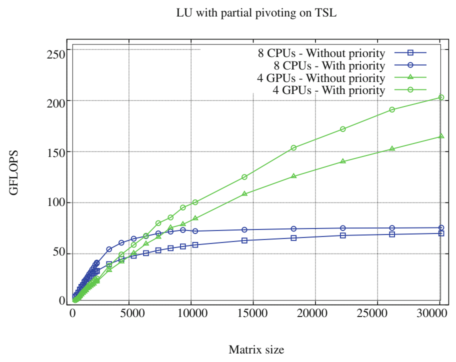


Fig. 5. Impact of the use of priority tasks on the performance of the LU factorization on TSL, using 8 CPU cores (multicore mode) and 4 GPUs (multiGPU mode).

5 Energy-Aware Extensions to SuperMatrix

In this section we present the energy-saving techniques introduced in the runtime, evaluating their practical outcome on the execution of the LU factorization.

5.1 Energy-Aware Runtime

Let us start by rehearsing our two energy-saving techniques for the task-parallel execution of DLA on hybrid platforms. In the following we will continue using the LU factorization with partial pivoting as the guiding case study, but the ideas and techniques described next also apply to any other DLA operation.

In order to motivate these techniques, consider again the bottom trace in Fig. 4. In that execution, four control threads (running each on a CPU core) and four GPUs of the platform collaborate in the factorization as follows: Due to their complexity, tasks of type LU are executed by the CPU cores, while the remaining two types of tasks, Trsm and Gemm, are run on the GPUs. Idle time (marked in white color) corresponds to periods when both the CPU cores and the GPUs perform no useful work, but they are instead waiting for the completion of an event. The trace certainly shows that these periods occupy a significant fraction of the execution. This inactivity occurs mainly because dependencies may determine that, at a given moment, there are no tasks ready for execution. Additionally, note that when a GPU is working, the corresponding CPU core remains inoperative, waiting for the completion of the job.

The question that naturally arises is how to leverage these inactive periods to reduce energy consumption. Although current architectures incorporate dynamic voltage and frequency scaling (DVFS) [9], our experiments in [2] reveal that the gains attained by reducing the CPU frequency during the execution of compute-intensive DLA operations are small. On the other hand, Fig. 6 illustrates that we can do better, even for hardware like TSL where DVFS technology is not available. In particular, the figure reports the power usage of one and four

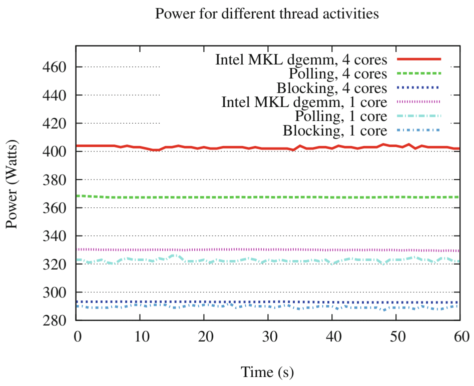


Fig. 6. Power consumption of different actions performed by threads.

cores of TSL, when repeatedly executing a matrix-matrix product (lines labelled as "Intel MKL `dgemm`"), performing a busy-wait (label "Polling"), and blocked (label "Blocking"). Interestingly, depending on the number of active cores, approximately 10–80 Watts can be saved by avoiding polling, provided this change does not increase the execution time. The results from this particular experiment justify the following two techniques introduced in SuperMatrix, that aim at replacing, busy-waits with a blocking power-friendly state whenever possible.

Avoid polling when there are no ready tasks. In SuperMatrix, when a CPU thread completes the execution of a task (and updates the corresponding dependencies), it queries the ready queue for more work, performing an active polling and, therefore, wasting energy in case there are no ready tasks. The goal of our first energy-aware technique is to avoid this situation, by employing instead a power-friendly blocking-wait (idle-wait).

For this purpose, we introduced POSIX semaphores into the runtime to control the activity of "idle" threads. In particular, when a CPU thread that polls the ready queue for a new task finds it empty (i.e., there is no task ready for execution at the moment), it blocks itself by invoking the system call `sem_wait()`. This requires some complementary mechanism to wake up blocked threads. In particular, when an active CPU thread completes the execution of a task and updates the dependencies, in case this implies moving c tasks from the work queue to the ready queue, this thread will also enforce that there exist c active threads, employing the system call `sem_post()`. This strategy ensures that there is basically one active thread per task in the ready queue and thus minimizes the introduction of delays in the execution of tasks while, at the same time, avoiding potential deadlocks.

Avoid polling when waiting for the GPU. When a CPU thread encounters a task of type TRSM or GEMM, to be executed on the binded GPU, it invokes the corresponding CUBLAS kernel. However, due to the asynchronous nature of the GPU kernels in NVIDIA CUBLAS, the calling CPU thread does not block, and so it queries the ready queue for more work. At this point, if the CPU thread dequeues a second task to be run in GPU, it tries to execute it, invoking the corresponding CUBLAS kernel. As a result, because in the SuperMatrix runtime the GPU only services one kernel at a time, the thread enters a busy-wait (polling) until the first kernel completes its execution on the GPU.

To avoid this behaviour, we modified the energy-aware runtime to invoke routine `cudaSetDeviceFlags` with the `cudaDeviceBlockingSync` parameter set, which blocks the CPU thread on a synchronization primitive when waiting for the device to finish work. We added the corresponding synchronization primitive after any call to CUBLAS performed within `libflame`. Routine `cudaSetDeviceFlags` allows to specify the behavior of the active host thread when it executes device code. After its activation, all synchronizations carried out using the primitive `cudaThreadSynchronize` will suspend the execution of the calling thread

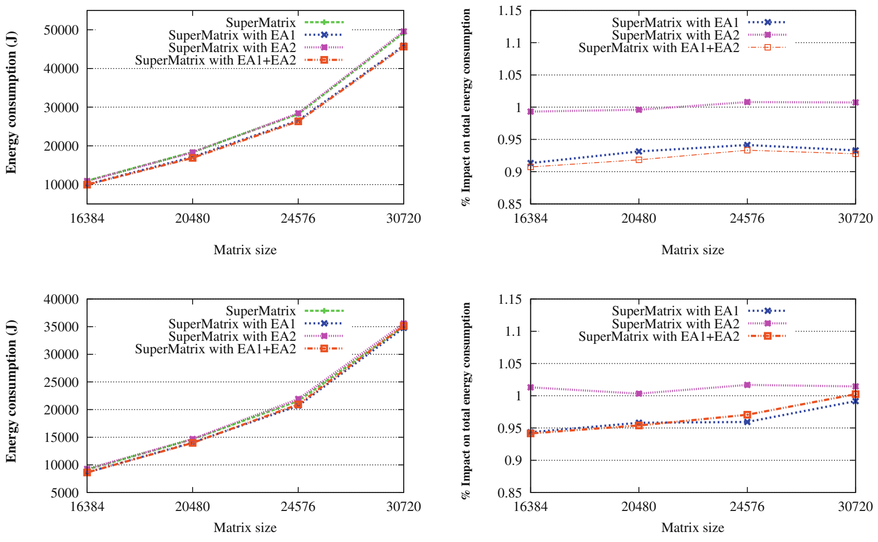


Fig. 7. Impact on energy of the energy-aware techniques of the LU factorization with partial pivoting without and with priority tasks (top and bottom, respectively).

until the device finalizes its work, thus blocking the core and avoiding the potential energy-waiting state.

Figure 7 reports the total energy consumption and energy savings attained for the LU factorization with partial pivoting when using a version of SuperMatrix without and with priority tasks, respectively. In all cases, we employed the multiGPU mode with 4 GPUs. In the plots, the label "EA1" denotes the first energy-aware technique, which avoids polling when there are no ready tasks; "EA2" stands for the second technique, which avoids polling when waiting for the GPU task completion. The combination of both techniques is referred to as "EA1+EA2". These results show a variety of energy gains, from close to 10% in some cases to a waste of energy in a couple of cases, depending on the DLA operation, matrix dimension, and technique.

The behavior of the power-aware runtime for the LU factorization with and without task priorities is particularly interesting. The reduction in execution time when task priorities are used yields an important reduction in energy consumption (compare the left plots in Fig. 7); however, this improvement in execution time is mainly motivated by a reduction in the amount of idle periods in the parallel execution. As our power-aware techniques exploit idle periods, the expected improvements from the application of these mechanisms are less significant as idle time decreases. That is the main reason that the energy savings are lower when priorities are applied (less than 6%) than when they are not (up to 9%). Finally, the results in Fig. 8 demonstrate that all these techniques introduce minimal overhead in the execution time.

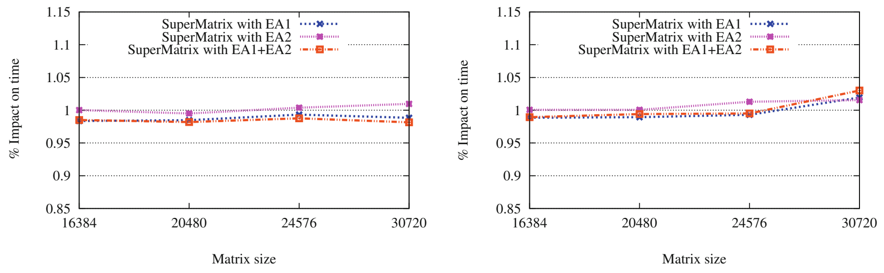


Fig. 8. Impact on execution time of the energy-aware techniques of the LU factorization with partial pivoting without and with priority tasks (left and right, respectively).

6 Concluding Remarks

In this paper we have introduced significant enhancements in the SuperMatrix runtime scheduler to address two different requirements of current HPC applications: the quest for high performance and the reduction of energy utilization. Although these two factors could be considered as orthogonal, our insights reveal mutual interactions that are illustrated using the LU factorization with partial pivoting, a well-known dense linear algebra operation, key to the solution of dense linear systems of equations.

On the performance side, we have introduced two different enhancements. First, a *hybrid execution* allows a better exploitation of the hardware resources available in current CPU/GPU platforms. Second, the adequate introduction and management of *priority tasks* yields a reduction in idle time and general performance improvement without intervention of the programmer. Our experiments reveal a performance speedup around 20% for the LU factorization with partial pivoting. These techniques and insights can be applied to other DLA operations in `libflame`, and also to other general-purpose runtime schedulers exploiting task parallelism.

With respect to energy, we have illustrated two different techniques that leverage idle times in parallel executions on hybrid CPU/GPU platforms to save energy. This approach yields fair power savings for a minimal impact on the execution time. Besides, these techniques are integrated into the runtime scheduler, and are transparent from the point of view of the developer.

Acknowledgments. This research was supported by project CICYT TIN2011-23283, MICINN-TIN 2008/508, TIN 2012/32180 and FEDER.

References

1. Alonso, P., Badia, R. M., Labarta, J., Barreda, M., Dolz, M. F., Mayo, R., Quintana-Ortí, E. S., Reyes, R.: Tools for power-energy modelling and analysis of parallel scientific applications. In: 41st International Conference on Parallel Processing - ICPP, pp. 420–429 (2012)

2. Alonso, P., Dolz, M. F., Igual, F. D., Mayo, R., Quintana-Ortí, E. S.: Saving energy in the LU factorization with partial pivoting on multi-core processors. In: Proceedings of the 20th Euromicro International Conference on Parallel, Distributed and Network-Based Processing - PDP (2012)
3. Alonso, P., Dolz, M. F., Igual, F. D., Mayo, R., Quintana-Ortí, E. S.: Reducing energy consumption of dense linear algebra operations on hybrid CPU-GPU platforms. In: Proceedings of the 10th IEEE International Symposium on Parallel and Distributed Processing with Applications- ISPA 2012, pp. 56–62 (2012)
4. Badia, R.M., Herrero, J.R., Labarta, J., Pérez, J.M., Quintana-Ortí, E.S., Quintana-Ortí, G.: Parallelizing dense and banded linear algebra libraries using SMPSs. Concurr. Comput. Pract. Exp. **21**(18), 2438–2456 (2009)
5. Barrachina, S., Castillo, M., Igual, F.D., Mayo, R., Quintana-Ortí, E.S., Quintana-Ortí, G.: Exploiting the capabilities of modern GPUs for dense matrix computations. Concurr. Comput. Pract. Exp. **21**(18), 2457–2477 (2009)
6. Bientinesi, P., Gunnels, J. A., Myers, M. E., Quintana-Ortí, E. S., van de Geijn, R. A.: The science of deriving dense linear algebra algorithms. ACM Trans. Math. Softw. **31**(1), 1–26 (2005)
7. Borkar, S., Chien, A.A.: The future of microprocessors. Commun. ACM **54**(5), 67–77 (2011)
8. Cilk project. http://supertech.csail.mit.edu/cilk/
9. Elnozahy, E.N., Kistler, M., Rajamony, R.: Energy-efficient server clusters. In: Falsati, B., Vijaykumar, T.N. (eds.) PACS 2002. LNCS, vol. 2325, pp. 179–197. Springer, Heidelberg (2003)
10. Esmaeilzadeh, H., Blem, E., St. Amant, R., Sankaralingam, K., Burger, D.: Dark silicon and the end of multicore scaling. In: Proceedings of the 38th Annual International Symposium Computer architecture, ISCA '11, pp. 365–376 (2011)
11. Igual, F. D., Quintana-Ortí, G., van de Geijn, R.: Scheduling algorithms-by-blocks on small clusters. Concurr. Comput. Pract. Experience (2012)
12. OmpSs project home page. http://pm.bsc.es/ompss
13. Paraver project. http://www.cepba.upc.es/paraver
14. PLASMA project home page. http://icl.cs.utk.edu/plasma
15. Quintana-Ortí, G., Quintana-Ortí, E. S., van de Geijn, R. A., Van Zee, F. G., Chan, E.: Programming matrix algorithms-by-blocks for thread-level parallelism. ACM Trans. Math. Softw. **36**(3), 14:1–14:26 (2009)
16. Quintana-Ortí, G., Igual, F. D., Quintana-Ortí, E. S., van de Geijn, R.: Solving dense linear algebra problems on platforms with multiple hardware accelerators. In: 14th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming - PPoPP, pp. 121–129 (2009)
17. StarPU project. http://runtime.bordeaux.inria.fr/StarPU/
18. Van Zee, F.G.: libflame. the complete reference, 2008. In preparation. http://www.cs.utexas.edu/users/flame

A Three Step Blind Approach for Improving HPC Systems' Energy Performance

Ghislain Landry Tsafack Chetsa[1,2](✉), Laurent Lefevre[1], and Patricia Stolf[2]

[1] INRIA, LIP Laboratory (UMR CNRS, ENS, INRIA, UCB), Ecole Normale Superieure de Lyon, Université de Lyon, Lyon, France
{ghislain.landry.tsafack.chetsa, laurent.lefevre}@ens-lyon.fr

[2] IRIT (UMR CNRS), University of Toulouse, 118 Route de Narbonne, 31062 Toulouse CEDEX 9, France
stolf@irit.fr

Abstract. Nowadays, there is no doubt that energy consumption has become a limiting factor in the design and operation of high performance computing (HPC) systems. This is evidenced by the rise of efforts both from the academia and the industry to reduce the energy consumption of those systems. Unlike hardware solutions, software initiatives targeting HPC systems' energy consumption reduction despite their effectiveness are often limited for reasons including: (i) the program specific nature of the solution proposed; (ii) the need of deep understanding of applications at hand; (iii) proposed solutions are often difficult to use by novices and/or are designed for single task environments.

This paper propose a three step blind system-wide, application independent, fine-grain, and easy to use (user friendly) methodology for improving energy performance of HPC systems. The methodology typically breaks into phase detection, phase characterization, and phase identification and system reconfiguration. And it is blind in the sense that it does not require any knowledge from users. It relies upon reconfigurable capabilities offered by the majority of HPC subsystems – including the processor, storage, memory, and communication subsystems – to reduce the overall energy consumption of the system (excluding network equipments) at runtime. We also present an implementation of our methodology through which we demonstrate its effectiveness via static analyses and experiments using benchmarks representative of HPC workloads.

1 Introduction

With the "race to exascale" one of the major concern for actors involved in the development and operation of HPC systems is no longer the number of PFlops (petaflops) their system can achieve per second, but how many PFlops they can achieve per Watt. This novel fashion of evaluating supercomputers' performance place a great emphasis on their energy consumption. This interest can be justified by the fact that computer chips seem to have hit a wall, meaning that we can't make them go any faster. Consequently, supercomputer designers just have to

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 168–181, 2013.
DOI: 10.1007/978-3-642-40517-4_15,

add more chips to increase computing power. But this approach has a significant impact on energy usage.

However, tremendous efforts are being undertaken by HPC operators from multiple levels to make supercomputers greener. This is evidenced by the Green 500 list; its latest issue shows that the greenest supercomputers are getting greener. The rise of graphics processors in massive server clusters and the acquisition of low power memories are probably the main reason of their sudden improvement in energy efficiency. Just to give a global picture, in 2010 Samsung claimed that more that 34TWh/year or $2.2B/year could potentially be saved if the memory in all 11.5Mu servers within the U.S. could be replaced with their Samsung Green DDR3 memory chip [1].

Similar efforts are being carried out regarding all other HPC subsystems from the processor to the network to the storage subsystems. However, significant efforts still need to be made if today's supercomputers want to meet the 20MW constraint for exascale.

There is a common believe that a considerable share of energy consumed by HCP systems during their operations could potentially be saved if user applications were programmed differently. Put another way, throughout their life cycle, user applications exhibit behaviours whose understanding allows implementing power reduction schemes which can significantly reduce the amount of energy they consume at runtime. This has been proven right by the literature [2–7].

From what precedes, making HPC applications more energy friendly requires designing or rewriting the applications with energy constraints in mind. These alternatives may not always be feasible. Rewriting some HPC applications is so costly that most people find paying the electrical bill worth (There is no evidence; however this issue has been in people's mind for a while, but to our knowledge no one has proposed an energy efficient version of an application so far.), whereas application developers usually don't pay much attention to how much energy their applications will consume. The main reason to this is that power saving schemes are platform specific. For example, let us consider the dynamic voltage and frequency scaling (DVFS) technology which allows scaling the processor's frequency according to the workload in some cases. Integrating DVFS into a program source code assumes that the developers know all the potential platforms that will run their applications which doesn't make sense. Although DVFS support is available in nearly all platforms today, at some point one need to select the appropriate frequency at which a specific must run. This can be very difficult to achieve at the coding stage since CPU frequency ranges are processor specific.

One could rely upon existing approaches such as those in the above references; unfortunately, they are application specific and require extensive knowledge from those. As a consequence, it can be extremely difficult or near to impossible to implement one of those approaches in your own HPC environment.

In this paper, we present a three step blind methodology for improving energy performance of HPC systems. The methodology typically breaks into *phase detection*, *phase characterization*, and *phase identification and system reconfiguration*. And it is blind in the sense that it does not require any knowledge from users. It allows system-wide, application independent, and reconfiguration of HPC subsystems including the processor, memory, storage, and communication subsystems. Its flexibility lies in the fact that it implements power saving schemes relying on computational behaviours also known as phases that the platform exhibits instead of those of individual applications. This flexibility enables its use on any HPC cluster provided that mechanisms upon which the power saving schemes rely are available on the platform. We also provide an implementation of our methodology through which we demonstrate its effectiveness considering benchmarks representative of HPC applications.

The remainder of the paper is organised as follows: Background and related work are presented in Sect. 2. In Sect. 3 we present our general purpose energy saving methodology. Section 4 presents a concrete implementation of the generic methodology for its evaluation along with experimental results. Finally, Sect. 5 concludes the paper and discusses future work.

2 Related Work

There is a large body of work addressing the issue of power consumption in high performance computing (HPC) systems. These work can roughly be divided into off-line and on-line approaches. Off-line approaches necessitating human intervention involve several steps including: source code instrumentation for performance profiling; execution with profiling; determination of the appropriate CPU-frequency for each phase; and source code instrumentation for inserting dynamic voltage and frequency scaling (DVFS) instructions. Freeh et al. [8] exploit PMPI to time MPI calls to insert DVFS scheduling calls based on duration while Cameron et al. [9] profile MPI communications. Kimura et al. [2] instrumented program source code to insert DVFS directives according to the program's behaviour in order to reduce the program's energy consumption without significant performance degradation.

On-line approaches attempt to detect program execution phases to apply DVFS accordingly. In [6,10] authors use on-line techniques to detect program execution phases, characterize them and set the appropriate CPU frequency accordingly. They rely upon hardware monitoring counters to compute runtime statistics – cache hit/miss ratio, memory access counts, retired instructions counts – which are then used for program phases detection and characterization. Policies developed in [6,10] tend to be designed for single task environments. The methodology we present herein bypass that limitation by focusing on the system instead of any individual applications. Its independence from any application allows its implementation on different systems without significant effort.

Online recognition of communication phases in MPI applications was investigated by Lim et al. [5]. Once a communication phase is recognized, authors apply CPU DVFS to save energy. They intercept and record the sequence of MPI calls during program execution and consider a segment of program code to be reducible if there are high concentrated MPI calls or if an MPI call is long enough. The CPU is then set to run at the appropriate frequency when the reducible region is recognized again.

Power saving schemes presented above are effective in the sense that they permit to reduce application's energy consumption without significant performance degradation; however, those techniques can hardly be used by non experts either because of the technique itself and/or because they sometimes require deep understanding of the application. For example, although intercepting MPI calls may be transparent, there is still the need to know what the application is doing in between those calls in order to set the appropriate frequency.

More recently in previous works [11,12], we showed that the energy consumption (the energy used when operating) of HPC systems can significantly be reduced through system reconfiguration mechanisms such as using DVFS to scale the processor's frequency down/up according to the workload. Those work can be seen as instances of the generic methodology we present in this paper; however, in this paper, we present a different phase characterization approach relying on the concept of last level cache per instruction ratio (LLCRIR). In addition, the phase identification approach also attempts to classify execution vectors in order to respond efficiently to phase changes.

3 Approach Description

HPC systems throughout their life cycle exhibit several behaviours – in terms of utilisation of resources (processors, memory subsystems, storage subsystems, and communication subsystems) – reflecting execution phases of a specific workload or workloads, which are similar in comparison with other workloads or regions of execution of a specific workload. Taking advantage of workloads variability and reconfigurable hardware, we propose a generic methodology for reducing the energy consumption of HPC systems. Our methodology breaks into three steps including: (i) *phase detection*, (ii) *phase characterization*, and (iii) *phase identification and reuse of configuration information*. It is labelled as "blind" because users do not need any information from workloads or applications being executed on the system. To guide the reader throughout this section, we have summarized the whole methodology in Fig. 1, where the phase detection step attempts to detect phases on a system which successively runs five different workloads (in this case, each workload is detected as a specific phase or behaviour the system went through).

3.1 Step 1: Phase Detection

The first step called phase detection is the process through which program/system phase changes are detected. A program phase is a region of execution throughout which a well defined metric is relatively stable. This definition assumes that performance is also relatively stable throughout a specific program phase or phase of execution of the program. Phase detection techniques fall into off-line and on-line techniques. Off-line phase detection techniques are irrelevant to our case since users only need to launch their applications and the methodology will "magically" do the work for them.

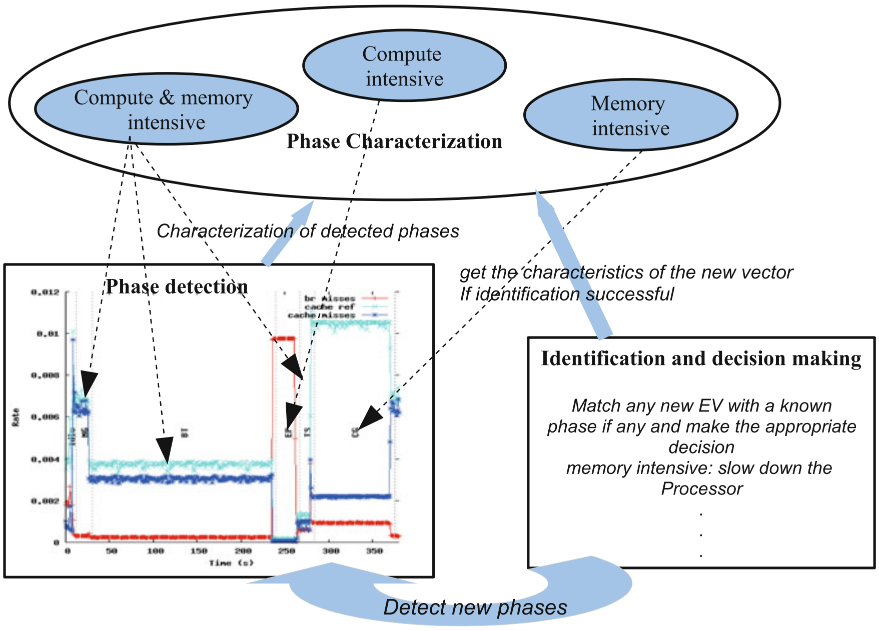

Compute intensive
Compute & memory intensive
Memory intensive
Phase Characterization
Characterization of detected phases
get the characteristics of the new vector
If identification successful
Phase detection
Identification and decision making
Match any new EV with a known phase if any and make the appropriate decision
memory intensive: slow down the Processor
Detect new phases

Fig. 1. Summary of the methodology on a system which successively runs five different workloads.

In general, phase detection mechanisms attempt to detect program phase changes, which sometimes require good understanding of the program itself. To avoid that, we suggest detecting phase changes at the system level. This makes sense because program phase changes are also reflected in the behaviour of the system (on which it is running) through resource utilisation. For example, when a program changes from a compute intensive/bound (we use the terms intensive or bound interchangeably) phase to a communication intensive phase, this also results in changes in utilization patterns of processor and communication subsystems. As the methodology is designed for non experts, the on-line phase detection mechanism must require nearly no user intervention. In addition, since phases are often too large for efficient representation and comparison in hardware, the detection step also involves compressing those phases (phase detection techniques usually use a few elements for representing and comparing phases).

3.2 Step 2: Phase Characterization

In the presence of dynamically reconfigurable hardware, initiating system reconfiguration at the right time is as important as selecting hardware or software eligible for reconfiguration (for power saving purposes, the most common reconfigurable hardware is the processor). As we mentioned earlier, a system goes through different phases or behaviours throughout its life cycle, so initiating system reconfiguration at the boundary of a phase seems natural; however, reconfiguring non eligible for reconfiguration hardware can result in significant performance degradation. The term "significant performance degradation" is a relative term and may be interpreted differently; however, a performance degradation of up to 10% is usually acceptable.

Our phase characterization process aims to associate each workload/phase with a label which implicitly indicates the type of system reconfiguration acceptable (which does not result in significant performance degradation) for that specific workload. We define five labels reflecting the kind of workloads a typical HPC system runs on a daily basis. These labels are: (a) *compute intensive*, (b) *memory intensive*, (c) *mixed*, (d) *network intensive* and (e) *I/O intensive*. They are self explanatory with the exception of "mixed". In a few words, workloads/phases labelled as mixed are both memory and compute intensive, which means that they alternate between memory intensive and compute intensive behaviours; however, the granularity at which this occurs is low to the point into which they cannot be considered as phases.

3.3 Step 3: Phase Identification and System Reconfiguration

Phase identification is the ability to identify recurring phases, or more generally to identify phases with each other. It is a desirable property for phase detection techniques, since it can be used in tuning algorithms to reuse previously found optimal configurations for recurring phases.

Phase identification is often used in conjunction with phase prediction. If the predicted phase is identified with an existing phase, then the optimal configuration (if there is any) for that specific phase is applied to the system. The coupling of phase identification and prediction for power/performance improvement will not be discussed in depth in this section because it widely depends on the phase detection technique.

Table 1 summarizes possible reconfiguration decisions that can be taken given a specific workload/phase label. Decisions are selected so as to guarantee that

Table 1. Phase labels and associated energy reduction schemes.

Phase label	Possible reconfiguration decisions
compute intensive	switch off memory banks; send disks to sleep; scale the processor up; put NICs into LPI mode
memory intensive	scale the processor down; decrease disks or send them to sleep; switch on memory banks
mixed	switch on memory banks; scale the processor up send disks to sleep; put NICs into LPI mode
communication intensive	switch off memory banks; scale the processor down; switch on disks
I/O intensive	switch on memory banks; scale the processor down; increase disks (if needed)

they do not result in significant performance degradation; they lie on the fact that some specific workloads might not need certain resources. Note that some elements in the table are counter-intuitive: switching on memory banks when running I/O intensive workloads is indeed efficient. An increase in RAM size reduces the dependency on disk which in turn improves the overall performance. If the system has several disks, some can be switched off instead of sending them to sleep, the reverse operation is performed if necessary when running I/O intensive workloads. Also notice that the disk (respectively the NIC) automatically changes to active when it is accessed.

4 Methodology for Reducing the Energy Consumption of HPC Systems: An Implementation

In this Section, we present an implementation of our generic methodology for reducing the energy consumption of HPC systems for its evaluation. There is a large body of work investigating phase detection techniques, so we will only provide an overview of the phase detection mechanism used in this work.

The phase detection technique we use relies upon the concept of execution vector (EV). An execution vector is simply a column vector of sensors including hardware performance counters, disk read/write and network byte sent/received counts. Sensors are selected so as to provide insight into resource utilization of the system. EVs are sampled on a per second basis and a phase change occurs when the Manhattan distance (which serves as a similarity metric) between two consecutive EVs exceeds a threshold which varies throughout the life cycle of the system. Note that phase changes are detected at the system level; meaning that we detect phases of the system. The number of EVs collected during a phase is proportional to its length; consequently, long run phases resulting in a huge amount of EVs cannot be efficiently stored. We address this by representing a phase with a single EV: the closest vector to the centroid of the group composed of EVs belonging to that phase. This vector is called reference vector.

4.1 Description of Our Phase Characterization Methodology

Unlike off-line phase characterization techniques, on-line phase characterization techniques must guarantee a minimal overhead on the host system. To characterize system phases or system behaviours at runtime, we rely upon memory sensitivity of workloads being executed. We define the memory sensitivity metric of a workload as its last level cache (LLC) references per instruction ratio (LLCRIR). A high LLCRIR indicates that the workload has stringent memory requirement while a low LLCRIR indicates that the workload is not memory intensive. Computing LLCRIR is as simple as reading two hardware events counters. Beside, modern processors have on-chip integrated facilities for counting events, so reading the counter can be done without any additional overhead.

We next associate characterization labels (labels are listed in Sect. 3.2) with phases or workloads according to the order of magnitude of the average LLC per

Table 2. Order of magnitude of LLC references per instruction ratio and associated labels.

Workload label	order of magnitude of LLCRIR
Compute intensive	$\leq 10^{-4}$
memory bound	$\geq 10^{-2}$
mixed (both memory compute intensive)	10^{-3}

instruction ratio of the corresponding phases. Table 2 defines the relationship between labels and the order of magnitude of LLC per instruction ratio averaged over the corresponding phases (figures in Table 2 are based on empirical evidences). As it can be seen from Table 2, the LLCRIR metric permits us to determine whether a workload is either compute intensive, memory intensive or a mixture of them. However, it does not tell the difference between communication intensive and I/O intensive.

This being an on-line power oriented workload characterization, a detailed workload characterization might be too costly. For characterizing I/O intensive workloads, we use the percentage of CPU time during which I/O requests were issued to any storage devices (bandwidth utilization for the device) as the I/O sensitivity metric; That percentage increases as the load on the disk increases; typically, a value close to 100% indicates that the disk is fully loaded. In this paper, we assume that a workload is I/O intensive when its disk utilization exceeds 50% (CPU time during which I/O requests are issued). We do not characterize network or communication intensive workloads; instead, we proceed by discrimination, meaning that, if a workload does not fall into a known and characterized group then, it is probably network intensive.

4.2 Phase Identification

As mentioned earlier herein, on-line system configuration algorithms often use phase identification together with phase prediction. The rationale behind predicting the next phase is the need to set up the appropriate system configuration at the boundaries of the ongoing phase before the new one gets started. Unfortunately, predicting the next phase without any information about the execution pattern of the workloads being executed can be very difficult if not near to impossible.

The similarity between execution vectors of recurring phases is likely to be very high. This said, we implicitly attempt to identify EVs with existing phases and take a reactive decision when the identification process is successful. The reactive decision lies on a principle widely used in caching algorithms; the idea is that if the system is running a task labelled as $label_1$ at time t it is likely to be running a task with the same label at time $t + 1$ (e.g., if the system is running a memory intensive workload/phase at time t, then it is likely to be running a memory intensive workload at time $t+1$). So to summarise, when an EV is identified with an existing phase having a label say $label_1$, then system configuration decisions associated to the corresponding label ($label_1$) are triggered; the same process is repeated for the next vector and so on. For example, if the system is running a memory intensive workload at time t (EV_t, execution vector sampled at time t is identified with a memory intensive phase/workload), then it is likely to be running a memory intensive workload at time $t+1$; consequently, at time t the system can be configured for running memory intensive workloads.

This works as long as EV_t is identified with an existing phase, but what happen when EV_t is unknown to the management mechanism (it is not identified with any known phase)? We assume that any unknown EV indicates a new type of workload and consequently a default system configuration can be defined for such cases. To mitigate the risk of degrading performance, we define the default configuration as the optimal system configuration (system configuration offering acceptable performance over a wide array of workloads) regarding performance.

4.3 Experiments and Results Analysis

Platform Description We evaluate our methodology on a 25 node cluster system (100 cores in total) set up on the French large-scale experimental platform called Grid5000 [13]. Each node is an Intel Xeon X3440 with 4 cores and 16 GB of RAM. Available frequency steps for each core are: 2.53 GHz, 2.40 GHz, 2.27 GHz, 2.13 GHz, 2.00 GHz, 1.87 GHz, 1.73 GHz, 1.60 GHz, 1.47 GHz, 1.33 GHz and 1.20 GHz. The Intel Xeon X3440 is provided with dynamic voltage and frequency scaling (DVFS) technology which allows users to scale its frequency and voltage in order to reduce the energy consumption of the processor. All the reconfiguration decisions are directed towards the processor since it is the sole component dynamically reconfigurable without extensive efforts (via DVFS) available to our evaluation platform.

Nodes are interconnected with Infiniband-20G and Linux kernel 2.6.35 is installed on each of them; perf event is used to read the hardware monitoring counters. Class B problem set of benchmarks – including Block Tri-diagonal solve (BT), Embarrassingly Parallel (EP), Conjugate Gradient (CG), Multi-Grid (MG), discrete 3D fast Fourier Transform (FT), Integer Sort (IS), and Scalar Penta-diagonal solver (SP) – from NPB-3.3 [14] is used for the experiments. During the experiments, NPB benchmarks use OpenMP as message passing interface (MPI) library.

The approach is implemented using two components both residing of each node of the cluster. The two components act in a client server like fashion. The client side captures resource utilisation metrics and performance counters, and implements system reconfiguration decisions; whereas the server side performs phase detection, phase characterization and identification. Decisions are taken locally to each node; however, the server side (available on each node) is capable of acting as a central server if a centralized decision maker is needed.

Experimental Methodology and Results As the methodology is designed to take advantage of varying workloads; the main objective is to see how the system reacts under different types of workloads. To this extent, we first define two basics system reconfiguration decisions targeting the processor. These decisions involve running workloads labelled as compute intensive at 2.53GHz, workloads labelled as memory intensive at 1.87GHz, and those labelled as mixed at 2.00GHz. These processor frequencies are not selected so as to guarantee that energy will be saved, but to make the impact of the methodology noticeable; however, they must be carefully selected in a production environment.

We next randomly executed workloads listed above several times to see how the system reacts in their presence. In a few words, EP and MG are compute intensive; BT, FT, and SP are mixed workloads and must fall into the mixed group at runtime; whereas, CG and IS are memory intensive workloads.

Initially, the processor's frequency on each node of the cluster is set to its maximum (2.53GHz in this specific case), which means that we first assume that all the workloads are compute intensive. This is arguable; however, we believe it guarantees a certain quality of service to workloads unbeknown to the management mechanism. With the frequency sets, we randomly execute each of the above workload five times (we selected five times because of the time constraint) while letting the system decides by itself the appropriate category of each workload. The system selects the label/category of the workload by setting the processor's frequency to the appropriate value. For example, for a compute intensive workload, the processor's frequency must be set to 2.53GHz, it is set to 2GHz and 1.87GHz for mixed and memory intensive workloads respectively. Since the characteristics of a workload are only known to the system when that workload has already been seen in the past (remember that the idea is to reuse configuration information for recurring phases/workloads) or when a similar workload is already known, we do not expect the first instance of any workloads to fall in the right category (or to be labelled by the most convenient label). Roughly speaking, the first instance of each workload serves as a reference point since it is likely to run at the highest frequency available.

Table 3 summarises decisions made by the system management mechanism. As expected, the first instance of each workload is considered compute intensive (CI in the table), because at that point the management mechanism does not have any information about them. Still from Table 3, we can notice that EP which is compute intensive was labelled memory intensive twice; however, the management mechanism redeemed itself for the 4th and 5th instances. Overall, Table 3 indicates that our methodology is capable of detecting, characterizing, and identifying recurring workloads or specific phases of a workload.

Figure 2 – where the legend from 1 to 5 represents the order of occurrence of each workload (1 for the 1th occurrence of the workload, 2 for the 2th, 3 for the 3th, 4 and 5 for the 4th and 5th occurrences respectively) – shows the impact of the management mechanisms on the energy consumption of recurring workloads. Figure 3 shows the execution time of each instance of the workloads at hand. Figures 2 and 3 indicates that on the system provided with energy reduction technologies, one can fully take advantage of our three step methodology to reduce the energy consumption of the overall computing infrastructure without significant performance degradation. On the one hand, the approach is capable

Table 3. Recurring workloads identification along with associated characteristics

Workload categories	Workloads	Instances 1st	2nd	3rd	4th	5th
compute intensive (CI)	MG	CI	CI	CI	CI	CI
	EP	CI	MI	MI	CI	CI
mixed (MIX)	BT	CI	MIX	MIX	MIX	MIX
	SP	CI	MIX	MIX	MIX	MIX
	FT	CI	MIX	MIX	MIX	MIX
memory intensive (MI)	CG	CI	MI	MI	MI	MI
	IS	CI	CI	MI	MI	MI

of reducing the energy consumption of some workloads such as MG with nearly no performance degradation (Fig. 3). On the other hand, for applications such as FT, BT, and SP the benefit in terms of energy reduction is less noticeable because of the increase in their execution time. However, the system was able to

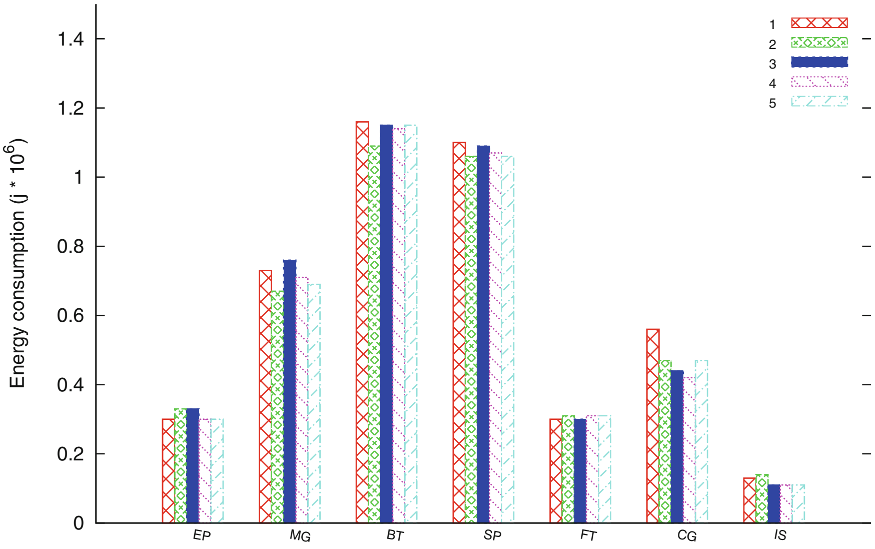


Fig. 2. Variations in energy consumption of recurring workloads due to decisions made by the management mechanism. The legend from 1 to 5 represents the order of occurrence of each workload (1 for the 1st occurrence of the workload, 2 for the 2nd, 3 for the 3rd, 4 and 5 for the 4th and 5th occurrences respectively).

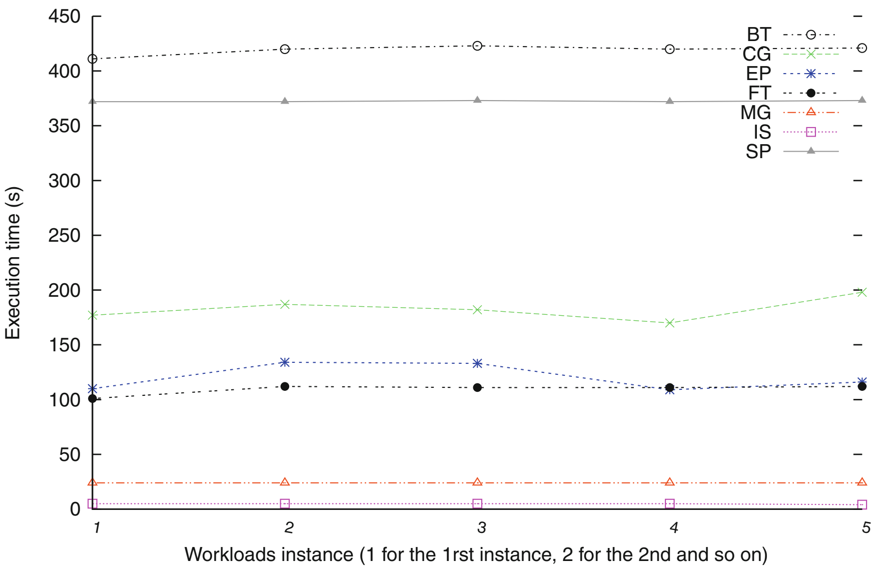


Fig. 3. Execution time of recurring workload with respect to reconfiguration decisions made by the management mechanism. The x-axis represents the order of occurrence of the workloads (1 for the 1st occurrence of the workload, 2 for the 2nd, 3 for the 3rd, 4 and 5 for the 4th and 5th occurrences respectively).

correctly label them; this suggests that their poor performance might be related to the CPU frequency at which they were run.

5 Conclusion

In this paper, we present a generic methodology to efficiently address the energy consumption problem of high performance computing (HPC) systems. It takes advantage of the variability of workloads that a typical HPC system runs on a daily basis, and breaks into three steps including (i) phase detection which attempts to detect system phases/behaviour changes; (ii) phase characterization which associate a characterization label to each phase (the label indicates the type of workload); (iii) finally, phase identification and system reconfiguration attempt to identify ongoing phases with known phases and make reactive decisions when the identification process is successful.

We further presented an implementation of the generic methodology and show how it can be used to effectively address the energy consumption in a system which experiences varying workloads. Results obtained with benchmarks representative of HPC systems show that in a HPC environment where power reductions technologies are available, our three step methodology can fully take advantage of those power reduction technologies to reduce energy consumption without any information about workloads being executed. As future work, we plan on improving the identification process in order to prevent erroneous labelling of workloads and extending the number of instances of workloads involved in the experiments. We are also planning to investigate more complex scenarios such as those wherein the system may experience idle periods; this implies an effective characterization of idle periods.

Acknowledgments. This work is supported by the INRIA large scale initiative Hemera focused on "developing large scale parallel and distributed experiments". Some experiments of this article were performed on the Grid5000 platform, an initiative from the French Ministry of Research through the ACI GRID incentive action, INRIA, CNRS and RENATER and other contributing partners (http://www.grid5000.fr).

It was also partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

References

1. Peng, S.: Green memory moving into the driver's seat. Intel Developer. Forum IDF (2010)
2. Kimura, H., Imada, T., Sato, M.: Runtime energy adaptation with low-impact instrumented code in a power-scalable cluster system. In: Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing, CCGRID '10, Washington, DC, USA, pp. 378–387. IEEE Computer Society (2010)
3. Freeh, V.W., Kappiah, N., Lowenthal, D.K., Bletsch, T.K.: Just-in-time dynamic voltage scaling: exploiting inter-node slack to save energy in mpi programs. J. Parallel Distrib. Comput. **68**(9), 1175–1185 (2008)
4. Rountree, B., Lownenthal, D.K., de Supinski, B.R., Schulz, M., Freeh, V.W., Bletsch, T.: Adagio: making dvs practical for complex hpc applications. In: Proceedings of the 23rd International Conference on Supercomputing, ICS '09, pp. 460–469. ACM, New York (2009)
5. Lim, M.Y., Freeh, V.W., Lowenthal, D.K.: Adaptive, transparent frequency and voltage scaling of communication phases in mpi programs. In: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing, SC '06. ACM, New York (2006)
6. Choi, K., Soma, R., Pedram, M.: Fine-grained dynamic voltage and frequency scaling for precise energy and performance tradeoff based on the ratio of off-chip access to on-chip computation times. Trans. Comp.-Aided Des. Integ. Cir. Sys. **24**, 18–28 (2006)
7. Ge, R., Feng, X., Cameron, K.W.: Performance-constrained distributed dvs scheduling for scientific applications on power-aware clusters. In: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing, SC '05, p. 34. IEEE Computer Society, Washington, DC (2005)
8. Freeh, V.W., Lowenthal, D.K.: Using multiple energy gears in mpi programs on a power-scalable cluster. In: Proceedings of the Tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP '05, pp. 164–173. ACM, New York (2005)
9. Cameron, K.W., Ge, R., Feng, X.: High-performance, power-aware distributed computing for scientific applications. Computer **38**, 40–47 (2005)

10. Isci, C., Contreras, G., Martonosi, M.: Live, runtime phase monitoring and prediction on real systems with application to dynamic power management. In: Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture, MICRO 39, pp. 359–370. IEEE Computer Society, Washington, DC (2006)
11. Tsafack, G.L., Lefevre, L., Pierson, J.-M., Stolf, P., Da Costa, G.: A runtime framework for energy efficient hpc systems without a priori knowledge of applications. In: ICPADS 2012: 18th International Conference on Parallel and Distributed Systems, Singapore, Singapore, pp. 660–667. IEEE, December 2012
12. Tsafack, G.L., Lefevre, L., Pierson, J.-M., Stolf, P., Da Costa, G.: Beyond cpu frequency scaling for a fine-grained energy control of hpc systems. In: SBAC-PAD 2012: 24th International Symposium on Computer Architecture and High Performance Computing, New York City, USA, pp. 132–138. IEEE, October 2012
13. Bolze, R., Cappello, F., Caron, E., Daydé, M., Desprez, F., Jeannot, E., Jégou, Y., Lanteri, S., Leduc, J., Melab, N., Mornet, G., Namyst, R., Primet, P., Quetier, B., Richard, O., Talbi, E.-G., Touche, I.: Grid'5000: a large scale and highly reconfigurable experimental grid testbed. Int. J. High Perform. Comput. Appl. **20**, 481–494 (2006)
14. Bailey, D.H., Barszcz, E., Barton, J.T., Browning, D.S., Carter, R.L., Fatoohi, R.A., Frederickson, P.O., Lasinski, T.A., Simon, H.D., Venkatakrishnan, V., Weeratunga, S.K.: The nas parallel benchmarks. The International Journal of Supercomputer Applications, Tech. Rep. (1991)

Performance Evaluation and Energy Efficiency of High-Density HPC Platforms Based on Intel, AMD and ARM Processors

Mateusz Jarus[1(✉)], Sébastien Varrette[2], Ariel Oleksiak[1], and Pascal Bouvry[2]

[1] Poznań Supercomputing and Networking Center, Noskowskiego 10, Poznań, Poland
[2] Computer Science and Communication (CSC) Research Unit, University of Luxembourg, 6, rue Richard Coudenhove-Kalergi, 1359 Luxembourg, Luxembourg
jarus@man.poznan.pl

Abstract. Due to growth of energy consumption by HPC servers and data centers many research efforts aim at addressing the problem of energy efficiency. Hence, the use of low power processors such as Intel Atom and ARM Cortex have recently gained more interest. In this article, we compare performance and energy efficiency of cutting-edge high-density HPC platform enclosures featuring either very high-performing processors (such as Intel Core i7 or E7) yet having low power-efficiency, or the reverse i.e. energy efficient processors (such as Intel Atom, AMD Fusion or ARM Cortex A9) yet with limited computing capacity. Our objective was to quantify in a very pragmatic way these general purpose CPUs using a set of reference benchmarks and applications run in an HPC environment, the trade-off that could exist between computing and power efficiency.

Keywords: Performance evaluation · Energy-efficiency · HPC · Intel Core I7 · Intel Xeon E7 · AMD Fusion · Intel Atom · ARM Cortex A9

1 Introduction

Many organizations have departments and workgroups that could benefit from High Perfomance Computing (HPC) resources to analyze, model, and visualize the growing volumes of data they need to conduct business. Up to now, most HPC systems are built on general purpose multi-core processors that use the x86 and Power instruction sets (both to ensure backward productivity and enhance programmers productivity). They are mainly provided by three vendors: Intel (around 71 % of the systems listed in the latest Top500 list[1]), AMD (12 %) and IBM (11 %). While initially designed to target the workstation and laptop market, these processors admittedly offer very good single-thread performance (typically 8 operations per cycle @ 2 GHz i.e. 16 GFlops), yet at the price of a

[1] Top500 List of November 2012 – see http://top500.org

DOI: 10.1007/978-3-642-40517-4_16,

Table 1. TDP of the top 4 processors technologies present in the Top500 List (Nov. 2012)

Processor Technology	**Top500 Count**	**Model Example**	**max. TDP**	
Intel Nehalem	225 (45%)	Xeon X5650 6C 2.66GHz	85W	14.1W/core
Intel Sandybridge	134 (26.8%)	Xeon E5-2680 8C 2.7GHz	130W	16.25W/core
AMD x86_64	61 (12.2%)	Opteron 6200 16C "Interlagos"	115W	7.2W/core
IBM PowerPC	53 (10.6%)	Power BQC 16C 1.6GHz	65W	4.1W/core

relative low energy efficiency. For instance, Table 1 details the Thermal Design Power (TDP) of the top four processors technologies present in the latest Top500 list. In parallel, the main challenge opened to the HPC community remains the building an Exascale HPC system by 2019 while staying within a power budget of around 20 MW. As current measures within a typical blade server estimate that 32.5 % of its supplied power are distributed to the processor, some simple arithmetic permit to estimate the average consumption per core in such an EFlops system: around 6.4 MW would be dedicated to the computing elements, and we can quantify their number by dividing the target computing capacity (1 EFlops) by the one of the current computing cores (16 GFlops) thus leading to approximately 62.5×10^6 cores within an Exascale system. Consequently, such a platform requires a maximal power consumption of **0.1 W per core**. In order to achieve this ambitious goal, alternative low power processor architectures are required. In this context, two main directions are currently explored: (1) relying on General-Purpose Graphics Processing Unit (GPGPU) accelerators (such as Nvidia Tesla cards or IBM PowerXCell 8i) or (2) using the processors (ARM, Intel Atom etc.) primarily designed for the mobile and embedded devices market. GPUs offer a greater performance per watts than conventional CPUs – for instance the Nvidia Tesla M2090 cards present in the HPC platform of the University of Luxemborg (UL) feature 512 cores for a TDP of 225 W thus leading to 0.44 W/core. However the effective usage of such devices remain a challenge as it requires for the developers to acquire new programming models. On the contrary, there is a growing demand for tablets and smartphones featuring low cost and power-effective computing elements. Much of this market is currently controlled by ARM Ltd. Taking into account only the tablet market makes the numbers impressive – at the end of 2011 there were 71 million tablets running ARM processors. They are easy to use with the classical programming models and as of 2013, it is the most widely used 32-bit instruction set architecture in terms of quantity produced. However, Intel became a major rival in this domain by developing Intel Atom CPU. What is more, low-power processors started to move to data center servers, most of them currently equipped with Intel devices. With the constantly growing electricity use by data centers worldwide, it will be increasingly important to make the servers less expensive. AMD also entered this market, developing his own low-power devices.

ARM processors also started an entry into the PC and server market, thanks to the success of complete ARM-based computers such as Rasberry Pi [12] or

Pandaboard [10] Future ARM processors are expected to be much more powerful in terms of performance while still maintaining low power usage. In this perspective, the European Mont Blanc project [3] plans to use ARM CPUs combined with Nvidia GPUs to build a Petascale HPC system. More concretely, a few companies start to release high-density ARM enclosures targeting the HPC market. It is the case of the Boston Viridis SystemBoard recently acquired by UL which features 48 Server-on-Chip (SoC) devices (ARM A9 Cortex 4C 1.1 GHz) in a 2U enclosure with a 300W power supply. In this article, we propose to benchmark this cutting-edge HPC platform, and to compare its performance (whether as regards the computing performance or the energy efficiency) to other high-density systems featuring either some alternative energy-efficient processors (such as Intel Atom, AMD Fusion), or "regular" high performance Intel processors (Core i7 or E7). This article is organized as follows: Sect. 2 presents the background of this work and reviews related works. Section 3 describes the experimental setup used within this study. In particular, we will present the different cutting-edge platforms we compare, together with the benchmark workflow applied to operate these comparisons. Then, Sect. 4 details the experimental results obtained. Finally, Sect. 5 concludes the paper and provides the future directions.

2 Context and Motivations

Energy efficiency is already a major concern in the design of computer systems, especially in the design of Exascale systems. There are a few activities targeting this problem. The aim of the Mont-Blanc project [3], launched on 1st October 2011 is to design supercomputer from ARM processors, using 15 to 30 less energy than conventional High Performance Computing platforms. The current status of this project is a proof-of-concept cluster named `Tibidabo` based on NVidia Tegra2 SoC (featuring ARM Cortex A9 processors having 2 cores at 1 GHz frequency). The cluster offer 128 such nodes in 38U for a measured performance of 120 MFlops/W. We will see that our ARM-based enclosure (2U) permits to reach around 572 MFlops/W.

Similarly, the EuroCloud [2] project was focused on building ARM-based Server-on-Chip, integrating 3D DRAM to provide a very dense low-power server. It introduces 10 times improvement in cost- and energy-efficiency compared to state-of-the art servers.

The trend of utilizing large number of low-power processors to replace high-end CPUs is becoming more and more popular. Different experiments prove embedded processors provide significant power savings. Zhonghong Ou [18] conducted a research, in which a cluster consisting of four PandaBoard development boards with ARM-based Cortex A9 MPCore was compared against an Intel workstation with quad-core Core2-Q9400 processor. They proved that the energy-efficiency ratio of the ARM cluster against Intel workstation varies from 1.21 to 9.5 evaluating three different applications – in-memory database, web server and video-transcoding.

Padoi E. L. et al. [19] compared the values of two metrics between ARM Cortex A9, Intel Xeon E7 and Intel Xeon E5: Time-to-Solution (the time needed to achieve the solution for a scientific problem) and Energy-to-Solution (the amount of energy spent to achieve useful results). The experiments show that ARM poorly performed on the Time-to-Solution metric compared to Xeon. In case of the Energy-to-Solution metric no winner was indicated. However, in most cases Intel Xeon E5 was more energy-efficient than ARM CPU, while Intel Xeon E7 usually achieved worse results than ARM. They conclude that the use of ARM processors to build HPC systems is still questionable.

Similar experiments were conducted to test the energy-efficiency of low-power CPUs only. In [16] ARM Cortex-A8 and Intel Atom N330 are compared with respect to their performance and energy-efficiency. The results show that the Cortex-A8 provides significant power savings, while Intel Atom reaches a higher processing power.

These examples show that the research on energy efficiency of processors is very extensive and popular. Despite measuring pure technical parameters of CPUs, many actions are also taken to reduce the energy consumption of existing devices, usually involing some software-based solutions. For example, in [17] the authors propose a method to estimate energy consumption of computers, thus making it possible to monitor the power draw in real time and introduce new ways to save energy. Similar experiments were performed in [15] where the authors proposed an approach to dynamic power consumption of multi-core processors or in [14], in which a methodology to predict power consumption of servers in data centers is presented.

These studies motivate us to further explore the field of energy-efficiency. In this paper we would like to compare the performance and energy efficiency of high-performance processors and low power CPUs using a larger set of platforms and types of benchmarks. Beside analyzing raw performance and energy consumption of the CPUs, we would also like to compare the platforms by measuring the latency and bandwidth.

3 Experimental Setup

3.1 Considered HPC Platforms

It is our belief that future HPC systems developed from today's energy-efficient solutions used in embedded and mobile devices (ARM, Intel Atom etc.) are the most likely to succeed. As we saw previously, several studies independently analyze the expected performances of such systems. Here, we focus on recent high-density enclosures designed for the HPC market build on top of the latest cutting-edge processors. More precisely, an overview of the selected systems we compare in this article is provided in Table 2.

CoolEmAll RECS platform Resource Efficient Computing System (RECS) is a platform capable to condense capacity of several hundred of servers in high density rack. It consumes only 35 kW, reducing the operating costs by 75 %.

Table 2. Overview of the high-density HPC enclosures benchmarked in this study

Name	Location	Size	#cpus	#RAM	Processor		max TDP/proc	
`i7`	PSNC (Poland)	1U	18	288GB	Intel Core i7-3615QE@2.3GHz	8C	45W	5.63W/c
`atom64`	PSNC (Poland)	1U	18	36GB	Intel Atom N2600@1.6GHz	2C	3.5W	1.75W/c
`amdf`	PSNC (Poland)	1U	18	72GB	AMD Fusion G-T40N@1GHz	2C	9W	4.5W/c
`bull-bcs`	UL (Luxembourg)	8U	16	1TB	Intel Xeon E7-4850@2GHz	10C	130W	13W/c
`viridis`	UL (Luxembourg)	2U	48	192GB	ARM A9 Cortex 1.1GHz	4C	1.9W	0.48W/c

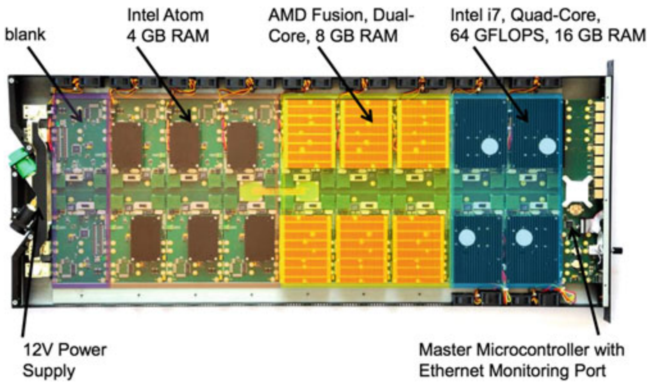

blank
Intel Atom 4 GB RAM
AMD Fusion, Dual-Core, 8 GB RAM
Intel i7, Quad-Core, 64 GFLOPS, 16 GB RAM
12V Power Supply
Master Microcontroller with Ethernet Monitoring Port

Fig. 1. CoolEmAll RECS platform – top view of one example rack featuring different processors

Every RECS unit has up to 18 energy efficient computing-nodes. The density of this approach is 4-10 times higher than blade servers. The main advantage of this system is the layered hardware infrastructure containing integrated sensors and micro-controllers for monitoring. It allows for fine-grained monitoring and management of every single node. With the use of additional visualization tools, such as 3D models of units in the rack (see Fig. 1), all of the system parameters can be easily accessed in real-time. The analysis of inlet and outlet air temperatures, whole node and CPU temperatures or system load leads to better energy efficiency.

This approach is used in the CoolEmAll project, coordinated by Poznan Supercomputing and Networking Center (PSNC). Within this project a range of tools is developed to enable data center designers and operators to plan and run facilities more efficiently. Currently there are 3 RECS units featuring different models of processors being monitored and benchmarked. All of them contain 18 nodes based on Intel Core i7-3615QE, Intel Atom N2600 or AMD G-T40N processors.

Each of the nodes in the first unit features quad-core Intel i7-3615QE@2.3 GHz processors [5]. However, thanks to the Hyper-Threading Technology there are 8 cores visible in the operating system. For each processor core that is physically present, the operating system can create two virtual processors and shares the workload between them [6]. What is more, the Turbo Boost Technology [8] allows it to automatically run the processor core faster than the marked frequency, up to

3.3 GHz. It is activated when the Operating System requests the highest processor performance state. It can also be enabled manually. In the experiments, additional tests were performed using this feature to measure the CPU performance and energy efficiency. The processor graphics is HD Graphics 4000, with the base frequency of 650 MHz. The available RAM memory on these nodes is 16 GB. The second unit contains Intel Atom N2600 processors, low-power, dual-core, 64-bit CPUs with Hyper-Threading Technology as well. They are built with 32 nm technology. The clock rate operates at a maximum value of 1.6 GHz. What is more, the processor comes with integrated graphics, with the base frequency of 400 MHz. Each node also features 2 GB of RAM memory. These devices are mainly targeted at tablets and netbooks. Finally, the last unit is equipped with AMD G-T40N processors [1]. It is an embedded platform that combines low-power, dual-core CPU and a discrete-level GPU into a single Accelerated Processing Unit (APU). It operates at a clock rate of up to 1 GHz. The available RAM memory is 4 GB. For the following tests, each of the nodes is running Linux 2.6.32, using the Scientific Linux distribution.

Boston Viridis platform featuring ARM Cortex A9 processors The Boston Viridis is a self contained, high-density 2U rack mount enclosure featuring 48 ultra-low power Server-on-Chip (SoC) based on ARM Cortex A9 processors delivered across 12 Calxeda EnergyCard modules and an integral high-speed 10 GbE interconnect.

The UL acquired two of these enclosures in January 2013 in a diskless configuration: the node system is provided over iSCSI by a management server. Due to the limited uplink of the enclosures (maximum 10 GbE yet limited for the moment to 1 GbE in our current setup), it makes no sense to benchmark this platform with applications that perform intensive I/O on the iSCSI disk. Indeed, whenever it was possible, we tried to use for I/O operation a partition mounted by `tmpfs`, a file system which keeps all files in virtual memory (thus limited in our case to 4 GB). Each system were deployed over a Ubuntu 12.10 ARM OS and a customized Linux kernel 3.5.0 (Fig. 2).

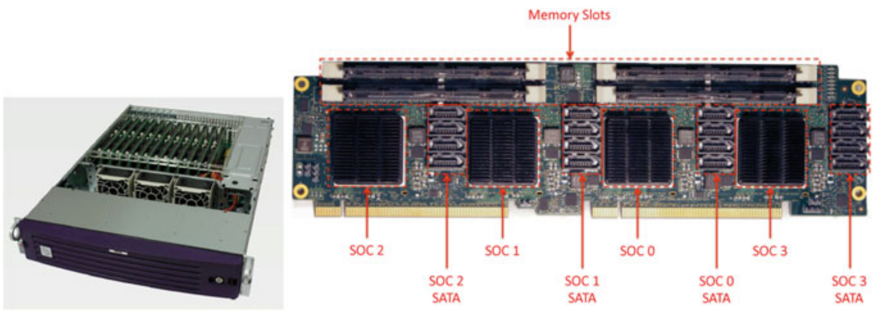


Fig. 2. A Boston Viridis enclosure: general overview and Calxeda EnergyCard modules.

Bull BCS platform featuring Intel Xeon E7 processors. Bull calls the BullX S6030 building blocks “Supernodes” because of the amount of memory that can be integrated in these servers. Each of these nodes feature 4 processors (Intel Xeon E7-4850@2 GHz 10C in the case of the UL configuration, with the frequency reaching 2.4 GHz in Turbo Mode) and a Bull-proprietary switch enables four of these nodes to work as one cache coherent Non-Uniform Memory Architecture (ccNUMA) system (the Bull BCS platform) with thus 160 cores (and 1TB of RAM). They are stacked with a 8U height configuration. The UL acquired this computing node in December 2012. It is part of the `gaia` cluster of the UL HPC platform, thus deployed over a Debian Squeeze (6.0.6) OS and a 2.6.32 Linux kernel.

3.2 Considered Benchmarks

Several reference benchmarks were selected to compare the platforms. For reproducibility reasons, all of them are open source. The following benchmarks were selected:

1. Phoronix Test Suite: it is a comprehensive testing and benchmarking platform, designed to carry out benchmarks in a clean and reproducible manner [11]. It includes a large set of different benchmarks, stressing system-wide components or selected hardware devices, such as disk, RAM or CPU. In the tests the following applications were used:
   – C-ray: this is a simple raytracing benchmark, usually involving only a small amount of data. This software measures floating-point CPU performance. The test is configured with a significantly big scene, requiring about 60 s of computation but the resulting image is written to /dev/null to avoid the disk overhead.
   – Hmmer: it is a software for sequence analysis. Its general usage is to identify homologous protein or nucleotide sequences. This type of problem was chosen because it requires a relatively big input size (hundreds of MB) and requires specific types of operations related to sequences.
   – Pybench: it offers a standardized way to measure the performance of Python implementations. In the past it has been used to track down performance bottlenecks or to demonstrate the impact of optimization and new features in Python. In contrast to the other benchmarks, it was run on one core only to test the power profile of servers running single-threaded applications.
2. CoreMark: it is intended to measure the performance of CPUs, using a set of algorithms: list processing, matrix manipulation or CRC. It is a single-threaded application (in the tests in was executed on one core only).
3. Fhourstones: this integer benchmark solves positions in the game of connect-4, as played on a vertical 7×6 board. The result is presented in fhourstones, where a fhourstone is taken as a thousand positions searched per second. It is a single-threaded application (in the tests in was executed on one core only).

4. Whetstone: synthetic floating-point benchmark, outputs the number of Whetstone Instructions per Second (WIPS). It is a single-threaded application (in the tests in was executed on one core only).
5. Linpack: floating-point benchmark for performing numerical linear algebra, outputs the number of floating-point operations per second (FLOPS). It is a single-threaded application (in the tests in was executed on one core only).
6. OSU Micro-Benchmarks: which permits to efficiently measure performance of MPI put and get operations.
7. High-Performance Linpack (HPL): the reference benchmark on the Top500 project. It solves a random dense linear system in double precision arithmetic on distributed-memory computers.

3.3 Performance and Energy-Efficiency Comparison

The aim of the first set of tests was to numerically compare the performance of all analyzed systems. For this reason the CoreMark, Fhourstones, Whetstones and Linpack benchmarks were ran, resulting in the benchmark specific values (as described in Sect. 3.2). To measure the impact of frequency on the final results, different experiments were performed on all available frequency values. Figures in Sect. 4 show averaged values over 100 tests for each frequency value. Because each CPU features different set of frequencies, Performance per MHz (PpMHz) was calculated for every experiment, represented as benchmark result divided by the frequency value on which the test was executed. What is more, Performance per Watt (PpW) was also calculated. For this reason, all of the benchmarks were executed for 120 s with the power usage of the CPU being constantly monitored. After the execution the raw benchmark result was divided by the average power draw, giving PpW value. Different results were achieved with different CPU frequency values. In the figures with the results (see Sect. 4) PpW metrics were presented for the highest frequency value. Only in case of Intel i7 and E7 processors two results are visible – one with the Turbo Mode enabled and the other one without it, running with the highest clock rate. The two values give insight into the performance of the CPU when compared to its frequency or wattage. In the second type of tests, benchmarks C-ray, Hmmer and Pybench were executed on all of the platforms with the same execution parameters. At the same time the power draw of the server and the execution times of the applications were monitored. The experiments were repeated for the following frequency values:

- ARM: 1.1 GHz,
- Intel Core i7: 1.2 GHZ (min), 2.3 GHz (max) and 2.31 GHz (max + Turbo Mode),
- Intel Atom: 0.6 GHz (min) and 1.6 GHz (max),
- AMD Fusion: 0.8 GHz (min) and 1.0 GHz (max),
- BCS: 1.064 GHz (min), 1.995 GHz (max) and 1.996 GHz (max + Turbo Mode).

In all of the platforms the minimum and maximum frequency values were tested. In case of ARM processor only 1.1 GHz was available. Turbo Modes were tested separately in case of Intel processors as the CPU consumes more power due to higher frequency.

The analysis of the power profile of presented platforms requires the measurements of machine's power usage under different system load. In these experiments the applications from the Phoronix test suite were executed, as described in Sect. 3.2. Different frequency values were analyzed to measure their impact on final results. All of the runs were executed 100 times to ensure that the results are statistically significant. The global workflow of the experiments is as follows:

1. t_0: Limit the CPU frequency to one of the previously described values (see Sect. 3.3) and start monitoring of the power usage and other system parameters (performance counters, memory usage, CPU temperature).
2. $t_0 + \Delta s$: Start one of the benchmarks.
3. t_1: Benchmark finished execution.
4. $t_1 + \Delta s$: End of monitoring.

The t_1 value is different for each benchmark, as every application takes different amount of time to finish. For the same platform it may be also different for different CPU clock rates, as it is also dependent on the frequency value. The Δ value was set at 30 s. Empirical tests proved that one minute (30 s after and before application execution) is enough to cool down the CPU after previous test. It is also the time required to gather the power profile of the platform in idle phase (power usage averaged over the range t_0 and $t_0 + \Delta$). The energy required to execute the application between $t_0 + \Delta$ and t_1 was also computed and compared between the platforms. In case of the ARM processors the power usage of one node had to be averaged over the four consecutive nodes. The reason for this is that the first node has a 12 V voltage regulator which is shared across all of the nodes on the energy card. As a consequence the power usage of the first node appears to be much higher than the rest. To get the exact power draw, the same application had to be executed on all nodes and the power usage divided by four.

4 Experiments

The experiments were divided into a few stages and include the above mentioned benchmarks i.e. the OSU Micro-Benchmark suite (which measures the Mesage Passing Interface (MPI) latency and bandwidth), CoreMark, Whetstones, Fhourstones, Linpack and High-Performance Linpack (HPL). We have performed these tests on the considered platforms and present now the results obtained, either from a pure performance point of view or from an energy-efficiency perspective.

OSU Micro-Benchmark 3.8 results. Despite the fact that we performed all the possible tests available in the OSU Micro-Benchmarks suite, we only present here the results of two Point-to-Point MPI benchmarks measuring the latency and the bandwidth that we consider as the most relevant tests from an HPC perspective. More precisely, for each considered platform, the following benchmark results are presented.

- `osu_get_latency` (Latency Test for Get with Active Synchronisation). Post-Wait/Start-Complete synchronisation is used in this test. The origin process calls `MPI_Get` to directly fetch data of a certain size from the target process's window into a local buffer. It then waits on a synchronisation call (`MPI_Win_complete`) for local completion of the Gets. The remote process waits on a `MPI_Win_wait` call. After the synchronisation calls, the target and origin process are switched for a message in the opposite direction. Several iterations of this test are carried out and the average get latency numbers is obtained. The latency includes the synchronisation time.
- `osu_get_bw`: (Bandwidth Test for Get with Active Synchronisation). The workflow is nearly the same as the one described for the latency estimation and thus also relies on Post-Wait/Start-Complete synchronisation.

The benchmarks results are presented in Fig. 3. At the level of the pure MPI performance, the worst results are obtained on the RECS platform featuring ATOM64 processors, while (as expected) the best ones are performed by the Bull BCS computing node. This is mainly due to the fast interconnect technology (InfiniBand QDR) present on the BCS platform. The energy profile has been tracked and is proposed in Fig. 4. It was not possible to draw the precise power profile of the BCS platform: both the latency and the bandwidth test finished in a time shorter than the frequency of our IPMI-based instrument for measuring the power consumption (1 measure every 20 s). It is indeed possible to estimate a lower bound of the energy used for each tests by tacking the idle power profile of this machine (2300 W).

CoreMark, Fhourstones, Whetstones and Linpack benchmark results. Raw benchmark values from all platforms are presented in Fig. 5. The graph

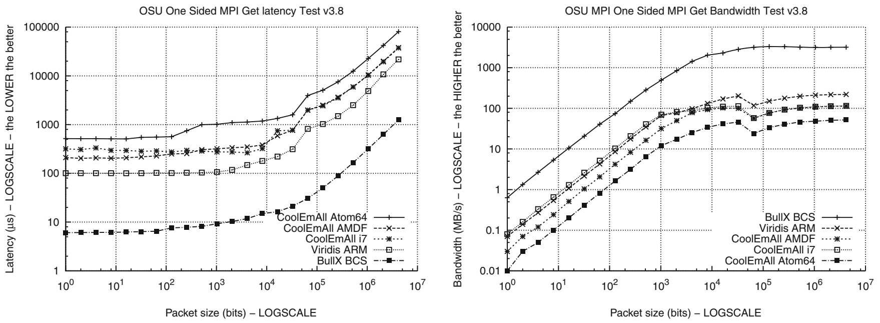


Fig. 3. OSU micro-benchmarks 3.8 performance results on the considered platforms.

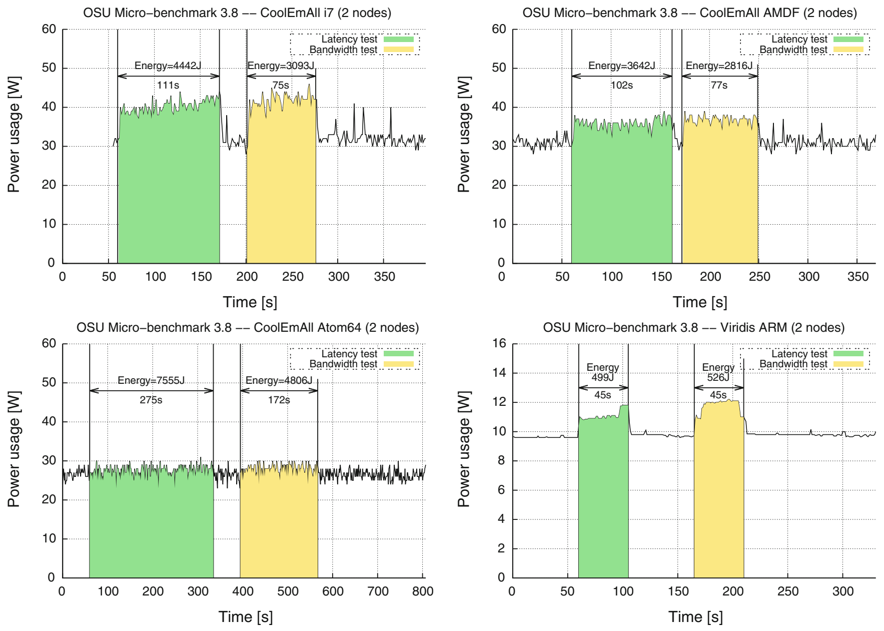


Fig. 4. OSU micro-benchmarks 3.8 – Cumulative power consumption (of the involved two nodes).

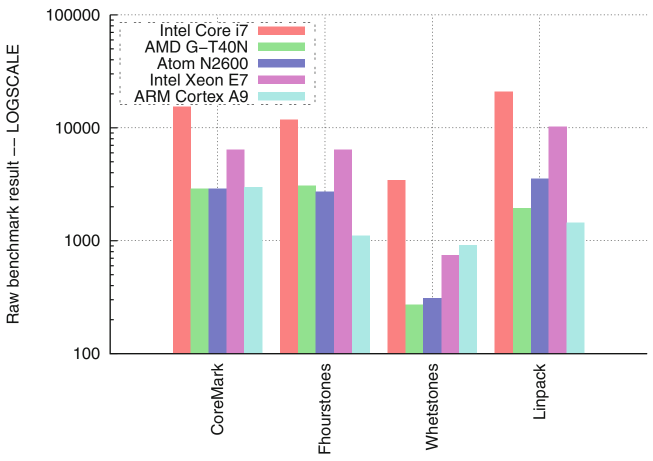


Fig. 5. CoreMark, Fhourstones, Whetstones and Linpack raw performances.

clearly shows that in all cases the best results are obtained by Intel Core i7. Intel Xeon E7 usually takes second place. In case of Whetstones application its results are similar to the power-efficient CPUs. AMD, ARM and Atom achieve comparable results, with a different leader for each benchmark.

Table 3. Best HPL results obtained on the considered platforms.

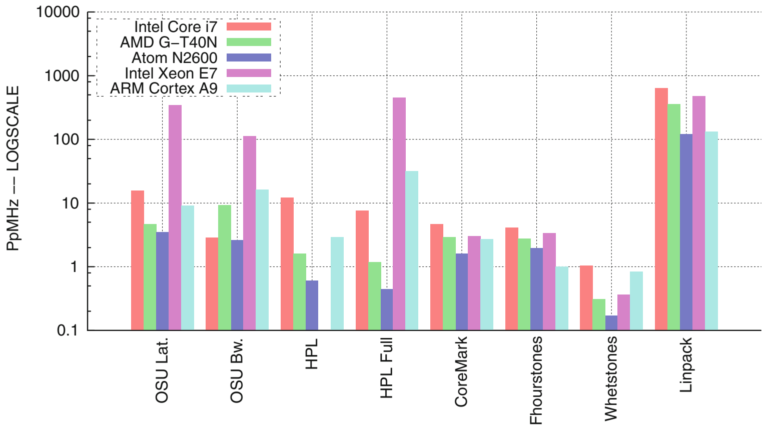
			Best HPL results								
Name	`#cpu`	R_{peak}	**N**	**NB**	**P**	**Q**	**Time [s]**	**GFlops** Effic.	**Energy[J]**	**PpMHz**	**PpW**
`i7`	1	73.6	41185	96	2	4	1175.15	39.63 53.85%	43976	12.009	1059.1
`amdf`	1	8	19496	160	1	2	3071.36	1.609 25,14%	57912	1.609	85.3
`atom64`	1	6.4	12891	112	2	2	1491.84	0.9575 11,97%	20671	0.598	69.1
`bcs`	1	80	n/a								
`viridis`	1	4.4	20711	96	2	2	1840.87	3.218 73.14	9983	2.925	593.7

Full platforms runs

Name	`#nodes`	R_{peak}	**N**	**NB**	**P**	**Q**	**Time [s]**	**GFlops** Effic.	**Energy[J]**	**PpMHz**	**PpW**
`i7`	18	1324.8	174733	96	12	12	7867.53	452.1 34.25%	6338465	7.61	561.163
`amdf`	16*	128	77984	160	4	8	16770.58	18.85 18.41%	4818744	1.1781	65.603
`atom64`	18	115.2	54692	112	8	9	8547.09	12.76 8.86%	1994357	0.4431	54.685
`bcs`	1	1280	87920	112	10	16	15115.57	1072 83,75%	50363322	446.6667	321.740
`viridis`	12*	52,8	63774	96	6	8	5090	34.39 65.14%	n/a	31.26	572.34

HPL Benchmark results. Of course, HPL is the reference benchmark to evaluate an HPC platform. Initially, we planned to perform the extended HPCC benchmark suite (which includes HPL) yet the support of the ARM platform for this suite is still pending. Note that in this article, due to the heterogeneity of the hardware environment and the focus on energy-efficiency, we performed here most of our HPL runs using a version built on top of the GNU compiler suite, with the notable exception of the BCS platform over which the Intel compiler suite was used. In practice, we have first performed the HPL benchmark on a single node i.e. a single CPU to quantify the optimized parameters `N`, `NB`, `P` and `Q`. Traces of theses successive runs are proposed in Appendix A, Fig. 8. This has been performed for every platform except the BCS one for which the vendor (Bull) provided optimized HPL parameters when the machine was initially configured. Then, based on the best run obtained so far (or the assessed optimized parameter for the BCS node), a full run involving the full platform has been performed. Table 3 summarizes the results obtained (also displayed in the Figs. 6 and 7). We put also in Appendix A some traces of the HPL runs performed to evaluate the five platforms. More precisely, the interested reader will find in the Fig. 8 the successive evaluations of various HPL parameters to find the ones that would hopefully lead to the best performance in more suited for a full run on all involved nodes... As for the full runs, we only present the power profile obtained in the Fig. 9. There are several interesting issues raised by these analysis. First of all, the efficiency of ARM platform is quite good (up to 73 %) despite the usage of the GNU compiler suite. At this level, terrible performances were obtained on the AMDF and Atom platform, where probably the usage of more appropriate compiling tools (the x86 Open64 Compiler Suite for AMD and the Intel Compiler suite for Atom) would raise better results. The results on i7 are conform to what was expected. And the BCS machine confirm its outstanding computing power, greatly helped by optimized HPL binaries[2].

[2] For licensing reasons, it was not possible to use Intel compiler on PSCN's platforms.



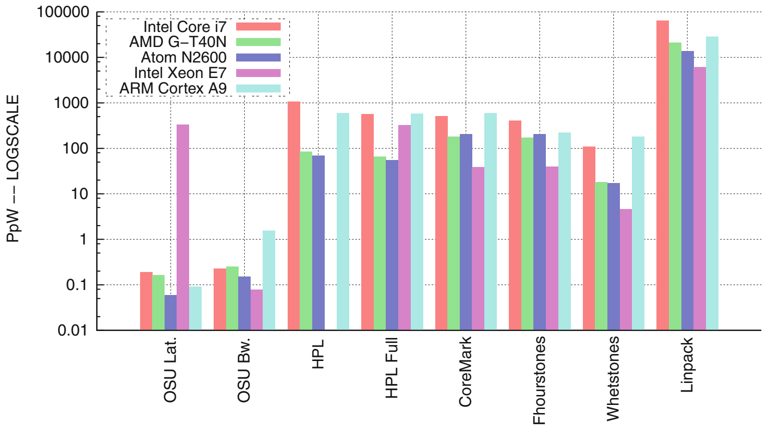


Fig. 6. PpMHz and PpW running all of the benchmarks on tested platforms

As regards the energy efficiency, the profile of both single and full HPL runs are expounded in Appendix A, in the Figs. 8 and 9. We can see that there was obvious issues with the reported power consumption of the atom nodes, which is still under investigation. Otherwise, the BCS platform demonstrates again its power-hungry profile yet a fair comparison (also at the level of the pure performance) should be performed on the basis of the PpMHz and PpW metrics, which is the purpose of the next section.

Comparison of benchmarks performance and energy-efficiency. On first sight it seems that the most performance-efficient CPU is Intel Core i7. However, it may be more accurate to compare the values resulting from the division of benchmark results by the frequency or power of the CPU. This results in relative values, which are more likely to compare between so diverse platforms. PpMHz and PpW results are presented in Fig. 6. There is a logarithm scale on Y axis, so

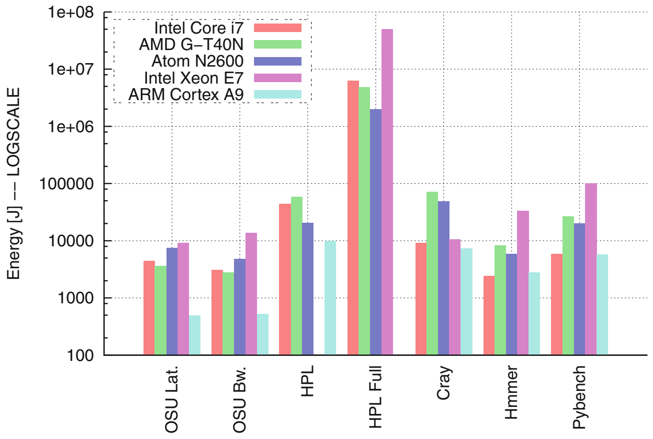


Fig. 7. Energy-efficiency of the executed benchmarks on the considered platforms.

Table 4. CoreMark results on Intel Core i7.

Frequency [MHz]	Raw result	PpMHz	Frequency [MHz]	Raw result	PpMHz
3300	15305.73	4.64	2300	10646.23	4.63
2200	10228.09	4.65	2000	9285.18	4.64
1800	8366.93	4.65	1600	7434.44	4.65
1400	6497.00	4.64	1200	5568.21	4.64

on first sight the differences in a few cases might seem very insignificant, although in reality larges differences in absolute values might be observed. The PpMHz values should remain constant and be independent of the used frequency. The tests proved this ratio to actually remain constant. For example, Table 4 presents raw benchmark values and the evolution of the PpMHz metric for different CPU frequencies. The Turbo Mode was discarded in tests because the frequency of the CPU could change during the tests in this state, adapting to current system load. The higher the value of the PpMHz metric, the better the performance per MHz of a given CPU.

In general, it is clearly visible that in the segment of high-performance processors, each processor justifies its targeted market. Intel Core i7 performs better on the Phoronix test suite (which perhaps corresponds more to a desktop usage) while Intel Xeon E7 takes the first place in more HPC-oriented tests (HPL and OSU) with respect to PpMHz and PpW. Intel Core i7 results are 1.2 to 1.6 better in case of CoreMark, Fhourstones and Linpack benchmark, but this ratio increases to almost 2.9 when running Whetstones. Several factors could have an impact on this result. It is important to remember, that presented benchmarks were executed on one core only. Intel Xeon E7 should perform better than Intel Core i7 when running multi-threaded applications, due to its higher number of cores. What is more, its clock rate is lower than the one of Intel Core i7. In Turbo

Modes the difference between them is 0.9 GHz. There are many other factors that affect the performance of the CPU and clock rate is only one of them. The type of instruction set architecture could also have a significant impact of the results. Intel Xeon E7-4850 uses Streaming SIMD Extensions (SSE) while Intel Core i7 has AVX [4] support, which improves performance by the introduction of wider vectors and new instructions.

In the group of low-power CPUs the results differ from benchmark to benchmark. These processors come from three different vendors and each of them features different tools to improve efficiency of the computations, yet minimizing power usage. ARM A9 Cortex features the Thumb-2 Technology [13] that provides enhanced levels of performance, energy efficiency, and code density. The NEON Media Processing Engine [9] provides support for the ARM v7 Advanced SIMD and Vector Floating-Point v3 instruction sets. This CPU also features optimized level 1 cache. Intel Atom N2600 provides Intel Hyper-Threading technology that increases processor throughput. The enhanced Intel SpeedStep Technology [7] is also implemented, allowing the clock speed to be dynamically changed during application execution, thus reducing power consumption when running less-demanding programs. AMD G-T40 provides additional boost capability enabled by AMD Turbo Core technology. The combination of low-power CPU and discrete-level GPU provides high performance multimedia content in a power-efficient platform. The differences between platforms are clearly visible in the results of the benchmarks. For example, similar results are achieved on AMD and ARM processors running CoreMark benchmark (2.88 and 2.71, respectively), with the Atom results being 1.7 to 1.8 times worse. On the other hand, Linpack results prove AMD to achieve highest PpMHz from these three low-power CPUs (354.54), almost three times better than the ones obtained on ARM and Atom. As regards OSU and HPL ARM platform clearly outperform its low-power counterparts. Careful analysis of the results leads to the conclusion that Atom achieves the worst results in almost all of the benchmarks. In the case of Fhourstones application it performed only better than ARM.

Figure 6 also shows values of PpW metric. Just like in the case of the PpMHz metric, the higher the value of PpW, the more efficient is the CPU per Watt. When analyzing the PpW metric it is important to remember that in the *bull-bcs* platform is build on components featuring four Intel Xeon E7 processors, not just one as in the case of the previous platforms. The number of the CPUs does not affect the values of PpMHz metric because only one core is always used when running these benchmarks, with the 159 of the others being idle. However, the idle cores consume more energy on Intel Xeon than idle cores on other platforms. Therefore, the results of this platform are very poor when compared to other CPUs. The results clearly show that the ARM processor achieves the best results when running OSU Bandwidth, CoreMark and Whetstones benchmark, while Intel Core i7 scores best on Fhourstones and Linpack. Although Atom and AMD CPUs are very power-efficient, the difference in performance between these platforms and the others worsens the results of PpW. Taking into account the power draw of all four Intel Xeon E7 processors, bull-bcs achieves up to

24 times worse results than other platforms. However, limiting the calculations to just one processor, as suggested in the previous paragraph, makes the results comparable to AMD and Atom.

Figure 7 presents the total energy consumed during the execution of all of the considered benchmarks and applications on tested platforms. In case of C-ray application, the OSU benchmarks and the HPL runs, ARM turned out to be the most energy-efficient. Although it required much more time to finish the calculations, its power draw remained very low. Second place usually took Intel Core i7. When considering the latency and bandwidth, AMD G-T40N achieved similar results. When running Hmmer it was the most efficient to use Intel Core i7. ARM obtained very similar result, while AMD and Atom required 2.5 to 3.5 times more energy. Intel Xeon E7 turned out to be very energy consuming, 14 times more than Intel Core i7. Its result was even worse on Pybench application, requiring more than 17 times more energy than Intel Core i7. Here again Intel i7 and ARM achieved very similar results, 5868 and 5730 joules, respectively. AMD and Atom consumed 3.5 to 4.5 times more energy.

It seems that the more threads the application spawns and the more load it places on the cores, the more energy- and time-efficient it is to run it on Intel Xeon E7. Idle cores consume a lot of energy and not placing load on them only worsens the results. It is important to remember that in this case all four Intel Xeon E7 processors were taken into account (160 cores altogether). In case of Pybench it means that 159 of them are useless (in case of Intel Core i7 only 7 are not used). However, dividing the power draw of Intel Xeon E7 by four (in other words – considering only one CPU in the calculations) does not make it very power-efficient. When executing Pybench it still consumes more than four times more energy than Intel Core i7. In terms of energy consumed ARM device performs very well in all cases. Although it often required more time to execute the applications, its total energy consumption was always one of the lowest. So the ARM platform turned out to be not only energy efficient, but also very performance efficient which place it as a strong candidate for an HPC system alternative. However, Intel Core i7 was both very fast in the computations and very energy-efficient at the same, sometimes even better than ARM. This justifies the leading position of this processor in the HPC market.

Table 5. Summary of the best results obtained as regards the energy-efficiency.

Name	Processor Type	MFlops/W	Green500 Rank*
`viridis`	ARM A9 Cortex	572.34	133
`i7`	Intel Core i7	565.13	134
`bull-bcs`	Intel Core E7	324.85	186
`atom64`	Intel Atom N2600	55.00	476
`amdf`	AMD Fusion G-T40N	65.45	467

* Based on November 2012 list http://www.green500.org/

5 Conclusion

Presented results clearly show that the tested platforms are targeted to different applications. Intel Xeon E7 and Intel Core i7 perform especially well in case of multi-threaded programs, utilizing much CPU load. Their execution time on these two processors is very short, compared to the low-power CPUs: AMD, Atom and ARM. Therefore, when the execution time matters, it is always better to choose performance-efficient CPUs, such as Intel Xeon E7 or Intel Core i7. Their execution time was up to 117 shorter compared to low-power devices. On the other hand, their power draw is very high. The competition between power-efficient devices is fierce and there's no single winner on the field of computational performance. The results are dependent on the executed benchmark. However, when considering PpW metric ARM Cortex A9 achieves always the best results. Moreover, out of the three considered low-power CPUs, it executes the programs in the shortest period of time and its total energy consumption is the least, in some cases up to 12 times lower that the energy usage of the rest of the CPUs. Probably we should have pushed our investigation further to include alternative compiling suites that might increase the overall performances of the AMD and Atom platforms, yet we can still conclude that an ARM-based platform such as the Viridis system used in this study offer a more than credible candidate for an HPC environment by the very interesting trade-off between performance and energy efficiency. The Table 5 summarizes the best results obtained by our benchmarking tests as regards the Green500 (see http://www.green500.org) performance metric (in MFlops/W i.e. the PpW metric used in the experiments section) and the corresponding rank that would be raised (using the latest Green500 list of Nov. 2012). Definitively, with the expected improvement planned in the ARM roadmap, building an Exascale HPC platform on top of such processor makes sense.

Acknowledgments. The experiments presented in this paper were carried out using the HPC facility of the University of Luxembourg and Poznan Supercomputing and Networking Center. The research presented in this paper was partially supported by a grant from Polish National Science Center (under award number 636/N-COST/09/2010/0), the FNR INTER-CNRS-11-03 Green@cloud project and the COST action IC0804.

A Appendix: HPL Runs Details

See Figs. 8 and 9.

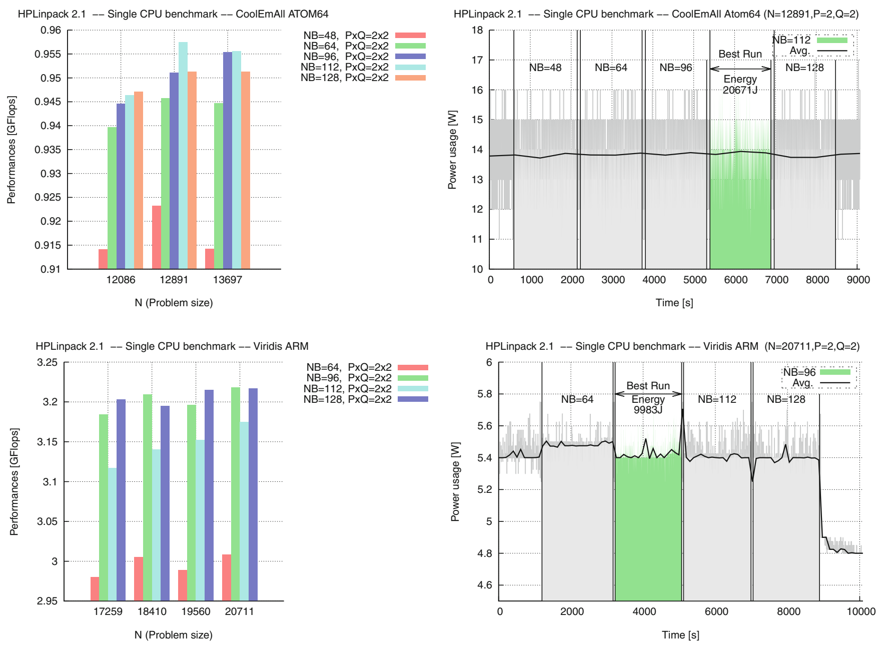


Fig. 8. HPLinpack 2.1 – Single CPU benchmark. Computing performances (left) and power consumption of the best runs (right) on the considered platforms.

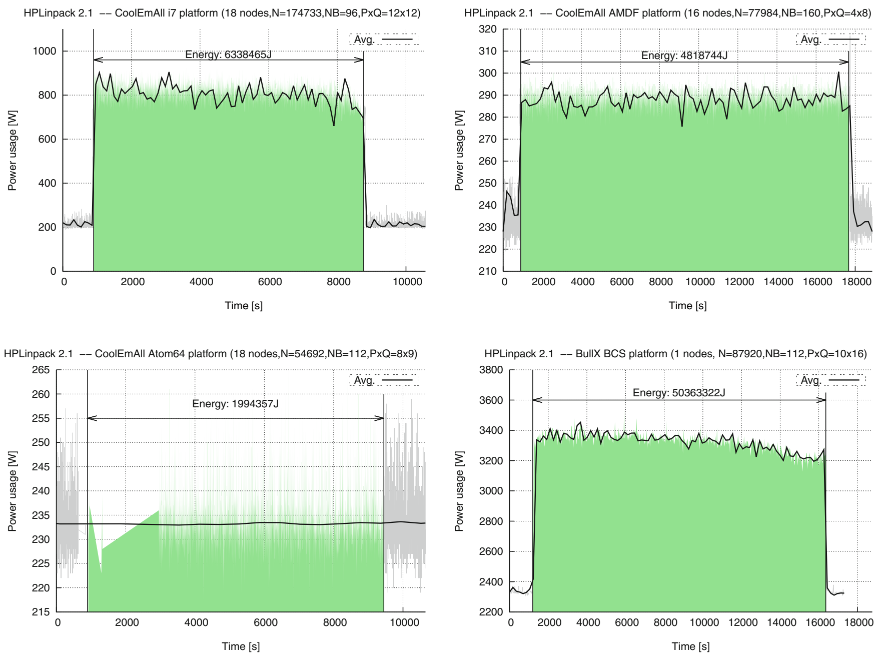


Fig. 9. HPLinpack 2.1 – Energy profile when performing the complete platforms benchmark.

References

1. AMD G-T40N. http://www.amd.com/us/products/embedded/processors/Pages/g-series.aspx
2. Energy-conscious 3D Server-on-Chip for Green Cloud. http://www.eurocloudserver.com/
3. European Mont-Blanc Project. http://www.montblanc-project.eu/
4. Intel Advanced Vector Extensions. http://software.intel.com/en-us/avx
5. Intel Core i7–3615QE. http://ark.intel.com/products/65709/Intel-Core-i7-3615QE-Processor-(6M-Cache-up-to-3_30-GHz)
6. Intel Hyper-Threading. http://www.intel.com/content/www/us/en/architecture-and-technology/turbo-boost/turbo-boost-technology.html
7. Intel SpeedStep Technology. http://www.intel.com/cd/channel/reseller/asmo-na/eng/203838.htm
8. Intel Turbo Boost Technology. http://www.intel.com/content/www/us/en/architecture-and-technology/turbo-boost/turbo-boost-technology.html
9. Neon Media Processing Engine. http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ddi0409i/index.html
10. PandaBoard. http://pandaboard.org/
11. Phoronix Test Suite. http://www.phoronix-test-suite.com
12. Raspberry Pi. http://www.raspberrypi.org/
13. Thumb 2 Technology. http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0471c/CHDFEDDB.html
14. Basmadjian, R., Ali, N., Niedermeier, F., de Meer, H., Giuliani, G.: A methodology to predict the power consumption of servers in data centres. ACM, New York (2011)
15. Basmadjian, R., de Meer, H.: Evaluating and modeling power consumption of multi-core processors. In: Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet, e-Energy '12, pp. 12:1–12:10. ACM, New York (2012)
16. Hoffman, K., Hedge, P.: ARM Cortex-A8 vs. Intel atom: architectural and Benchmark comparisons. Technical report, University of Texas at Dallas (2009)
17. Jarus, M., Oleksiak, A., Piontek, T., Weglarz, J.: Runtime power usage estimation of HPC servers for various classes of real-life applications. To appear in Future Generation Computer Systems (2013)
18. Ou, Z., Pang, B., Deng, Y., Nurminen, J., Ylä-Jääski, A., Hui, P.: Energy- and cost-efficiency analysis of ARM-based clusters. In: 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid 2012, pp. 115–123 (2012)
19. Padoin, E.L., de Oliveira, D., Velho, P., Navaux, P.: Time-to-solution and energy-to-solution: a comparison between ARM and Xeon. In: Third Workshop on Applications for Multi-Core Architectures (WAMCA), pp. 48–53 (2012)

Part IV

Wired and Wireless Networking

Enhancing IEEE 802.11 Energy Efficiency for Continuous Media Applications

Vitor Bernardo[1](✉), Marilia Curado[1], and Torsten Braun[2]

[1] Center for Informatics and Systems, University of Coimbra, Coimbra, Portugal
{vmbern, marilia}@dei.uc.pt

[2] Institute for Computer Science and Applied Mathematics, University of Bern, Bern, Switzerland
braun@iam.unibe.ch

Abstract. This paper proposes the Optimized Power save Algorithm for continuous Media Applications (OPAMA) to improve end-user device energy efficiency. OPAMA enhances the standard legacy Power Save Mode (PSM) of IEEE 802.11 by taking into consideration application specific requirements combined with data aggregation techniques. By establishing a balanced cost/benefit tradeoff between performance and energy consumption, OPAMA is able to improve energy efficiency, while keeping the end-user experience at a desired level. OPAMA was assessed in the OMNeT++ simulator using real traces of variable bitrate video streaming applications. The results showed the capability to enhance energy efficiency, achieving savings up to 44% when compared with the IEEE 802.11 legacy PSM.

Keywords: IEEE 802.11 · Energy efficiency · Power save mode

1 Introduction

The opportunity to connect mobile equipment, sensors, actuators and other devices to the Internet, usually referred as Internet of Things (IoT) [1], raises new challenges in the deployment of those equipments. The battery lifetime is still one of the most relevant challenges, since it is directly affected by the device communication capabilities. Despite numerous efforts to create alternative low power radio technologies, IEEE 802.11 seems to be the de facto standard for wireless communications in most common scenarios. Therefore, it is crucial to investigate and propose mechanisms aimed at saving energy while providing Internet access through an IEEE 802.11 ready interface.

Furthermore, the massive deployment of high demand continuous media application, namely Video on Demand (VoD) or Internet Protocol Television (IPTV), also enforces new requirements with respect to the equilibrium between energy efficiency and application performance. Besides specific application constraints other aspects may be considered, such as end-user guidelines about whether or not energy saving is mandatory. For instance, the end-user configuration can be related with daily mobility or traveling patterns. As the end-user battery lifetime expectations are extremely hard to predict, the inclusion of end-user feedback in the optimization process will bring some benefits.

This work proposes the Optimized Power save Algorithm for continuous Media Applications (OPAMA), aiming to improve devices' energy consumption using both end-user and application specific requirements, together with an optimized IEEE 802.11 power saving scheme and frame aggregation techniques.

The remaining sections of this paper are organized as follows. Section 2 discusses the related work, followed by the OPAMA proposal presentation in Sect. 3. The assessment of OPAMA performance, in the OMNeT++ simulator, is described in Sect. 4. Lastly, Sect. 5 presents the conclusions.

2 Related Work

This section introduces the background of the proposed algorithm, and presents the most relevant related work concerning IEEE 802.11 energy efficiency improvements for continuous media applications employing power saving techniques.

An IEEE 802.11 station (STA) under Power Save Mode (PSM) [2] (also known as Legacy-PSM) is able to switch off the radio during a certain period, aiming at saving energy during that time. A STA must inform the Access Point (AP) about the current power management mode by defining the corresponding power management fields in the control frames. When the power saving mode is enabled for a STA, the AP buffers all the packets to that station. If the AP has packets buffered to a certain STA, it will send a notification using the Traffic Indication Map (TIM) field of the *Beacon* frames. In the PSM, a STA must wake-up regularly to receive the *Beacon* frames. By performing this action, a STA that does not have any data buffered on the AP will be required to wake up recurrently, resulting in unnecessary energy consumption. To overcome this limitation, IEEE 802.11e [3] introduced the Unscheduled Automatic Power Save Delivery (U-APSD) algorithm. The main difference between the PSM and the U-APSD is related to the proactivity implemented in the U-APSD scheme. Unlike PSM, where only the Access Point (AP) is able to inform the station about pending packets, in U-APSD, the STA can itself ask the AP for new downlink messages pending in the queue. More recently, IEEE 802.11n [4] also announces two contributions to the power saving schemes, namely the Spatial Multiplexing (SM) Power Save and the Power Save Multi-Poll (PSMP) techniques.

Energy saving mechanisms for IEEE 802.11 can consider cooperation between the energy aware mechanisms at the lower (e.g. MAC layer aggregation) and upper layers. Camps-Mur et al. [5] have studied the impact of IEEE 802.11 MAC layer aggregation on both PSM and U-APSD schemes. The authors proposed a Congestion Aware-Delayed Frame Aggregation (CA-DFA) algorithm, which is divided into two logical parts: congestion estimation and dynamic aggregator. Congestion estimation is responsible for assessing the network capabilities and uses these values as near real-time input for dynamic aggregation. Being able to measure accurately network congestion, it allows the algorithm to dynamically adapt the maximum frame aggregation size when the network congestion goes below a certain limit. When compared with the IEEE 802.11 standard aggregation schemes, the CA-DFA performance is superior, particularly in terms of energy consumption. However, the CA-DFA algorithm does not support any end-user feedback.

Tan et al. [6] proposed a cross-layer mechanism based on the standard PSM, but using information provided by the upper layers. The algorithm, named PSM-throttling, aims at minimizing energy consumption for bulk data communications over IEEE 802.11. The PSM-throttling concept is based on the idea that there are already many Internet based applications performing bandwidth throttling and, as a result, there is an opportunity to improve energy efficiency at the client side. PSM-throttling uses the under-utilized bandwidth to improve the energy consumption of bandwidth throttling applications, such as video streaming. Nonetheless, it does neither consider the inclusion of dynamic aggregation, nor the possibility that the end-user controls himself the maximum allowed delay.

An adaptive-buffer power save mechanism (AB-PSM) for mobile multimedia streaming was proposed by Adams and Muntean [7] to maximize the STA sleep period. The proposal includes an application buffer, able to hide the frames from the Access Point and, consequently, to avoid the TIM reports with pending traffic indication. The authors argue that the amount of packets to store in that buffer could be dynamic, but they do not explain how to overcome this issue. Moreover, AB-PSM aims to be an application-based approach, but the mechanism to be used by the STA to provide feedback to the AP was not defined. Additionally, aggregation mechanisms were not employed and the testbed study is very limited, since only battery lifetime was analyzed. This is an important parameter, but it should always be correlated with the drawbacks introduced in the end-user application (e.g., extra delay or jitter).

According to Palit et al. [8] the feasibility of employing aggregation is strongly related with the scenario and/or application. In order to understand the typical packet distribution in a smartphone data communication, the authors have analyzed mobile device traffic. The main observations are that around 50% of the packets have a size less than 100 bytes and 40% have an inter-arrival time of 0.5 ms or less. These conditions enable a good opportunity to perform aggregation. Using this motivation, the authors have studied the aggregation impact in the smartphones' energy consumption. The proposed aggregation scheme uses a buffering/queuing system in the AP together with PSM in the client side. The proposed packet aggregation mechanism, named Low Energy Data-packet Aggregation Scheme (LEDAS), receives packets from the different applications through the Logical Link Control sub-layer and performs the aggregation. This approach showed some good results, but application requirements, such as the maximum tolerable delay, were not taken into account. With the native support for frame aggregation in IEEE 802.11n [9], which includes two distinct approaches to perform MAC frame aggregation, named Aggregated MAC Service Data Unit (A-MSDU) and Aggregated Mac Protocol Data Unit (A-MPDU), various studies concerning aggregation performance have been done [10]. Kennedy et al. studied the adaptive energy optimization mechanism for multimedia centric wireless devices [11] and concluded that significant energy saving could be achieved when performing application-aware optimization. Pathak et al. [12] have proposed an application level energy consumption profiling tool for mobile phones and reported issues concerning high energy usage in I/O operations. The software-based energy methodologies were early surveyed by Kshirasagar [13].

Although others in the literature [7,14] have also proposed energy optimization for continuous media applications none takes advantage of all the key optimization parameters proposed in this work. To the best of our knowledge, this paper proposes an original optimized power saving algorithm for continuous media applications, which combines the usage of buffering techniques and frame aggregation mechanisms, while using the end-user feedback to keep the application quality within the defined limits. Additionally, although the novel power saving modes and aggregation schemes are available in more recent IEEE 802.11 standards, the Legacy PSM still is the de facto standard algorithm concerning PSM in IEEE 802.11, while the implementation of other algorithms is mainly optional. As a result, the proposed algorithm is based on Legacy PSM and uses A-MSDU aggregation, which is already mandatory in the reception side of the IEEE 802.11n standard.

3 Optimized Power Save Algorithm for Continuous Media Applications

This section introduces the proposed Optimized Power save Algorithm for continuous Media Applications (OPAMA).

3.1 Motivation

Mobile end-user energy constraints are still one of the critical issues to be addressed in wireless communication protocols, particularly at the MAC Layer. IEEE 802.11, the most popular in real world equipment wireless technology uses the Power Save Mode (PSM), usually referred in the literature as Legacy Power Save Mode (Legacy-PSM), to limit energy consumption. However, the Legacy-PSM utilization in the presence of continuous media applications (e.g., video or voice) does not bring considerable energy savings, due to protocol design limitations, as explained next.

Legacy-PSM buffers traffic at the Access Point (AP) to all the stations (STA) operating in PSM mode, which indicate that they are in a *doze* state. A STA must wake-up to receive the *Beacon* sent by the AP at the beginning of each *Beacon Interval*. When broadcasting a *Beacon*, an AP supporting PSM must look for pending packets for each STA in a *doze* state that is currently associated with the AP. If there is data pending for a certain STA, the AP reveals it through the Traffic Information Map (TIM) field present in the *Beacon*.

When receiving a *Beacon*, a STA analyzes the TIM to verify the pending information existing in the AP buffer. Once there is pending data, the STA sends back a *PS-Poll* message to the AP asking for the data. The AP may reply with a single acknowledgement (ACK) or directly with the pending data frames. Then, the STA must stay awake while the *MoreData* flag is set. The AP will set this flag, while there is data to be delivered, while the STA should send back a *PS-Poll* for each pending frame. Therefore, when receiving data from a continuous media application, the STA will not be able to stay in a *doze* state for long, since there will be almost always some data to be received. As a result, even if the device battery is near a critical threshold, it will not be possible to save energy by employing the Legacy-PSM. A detailed discussion concerning PSM operation and buffer-related issues at the AP was performed by Zhu et al. [15].

OPAMA addresses these issues by introducing the end-user expected performance feedback in the process, allowing higher control opportunities at the AP. The next subsection presents OPAMA design and architecture.

3.2 Architecture

The main goal of OPAMA is to allow the end-user to save energy while keeping a desired quality at the application level. For instance, when the device battery is low, the end-user might like to have the possibility to slow down the transmission performance up to a certain level in order to save energy. To accomplish this goal, the STA sleep periods must be maximized. Consequently, OPAMA will manage the AP buffer differently compared to Legacy-PSM. While the Legacy-PSM will always inform the STA about any pending data to the STA, OPAMA will employ an algorithm based on the end-user expectations for the application performance to decide when pending data information should be sent to the STA. As on Legacy-PSM, OPAMA pending packets will stay in the AP queue. As a result, this operation will not affect the Legacy-PSM standard protocol [7].

Figure 1 depicts a simplified operation scenario of OPAMA. *STA-1* is operating in a doze state, and it is being served by *AP-1*, which is then connected to the core network (not represented here).

OPAMA operates as follows: *STA-1* left the *doze* state to receive *Beacon-1*. As there are no pending frames to be delivered, it just goes back into sleep mode. The first data for *STA-1* arrives at the *AP-1* when STA-1 is sleeping, then it is buffered. Again, *STA-1* becomes awake to receive *Beacon-2*. At this moment, there is already pending data for STA-1. However, OPAMA will employ a specific algorithm (Algorithm 1) to determine whether *STA-1* should be informed about pending data. In the example of Fig. 1, the algorithm returned false and the TIM of *Beacon-2* does not include information about pending data for *STA-1*. The pending data information is only sent within *Beacon-3*, followed by the data transmission start. Later, in *Beacon-5* OPAMA decides again to queue the frames for a longer time, allowing *STA-1* to return into the *doze* state with pending data available.

When the frames stay longer in the AP queue there are more opportunities to perform aggregation, as represented in Fig. 1. In this case, *Frame-1*, *Frame-2*, *Frame-3* and *Frame-4* were aggregated using the A-MSDU scheme into *Frame-A1* and *Frame-5*, *Frame-6* and *Frame-7* into *Frame-A2*. The number of frames present in each A-MSDU is dynamic and depends on the total amount of bytes

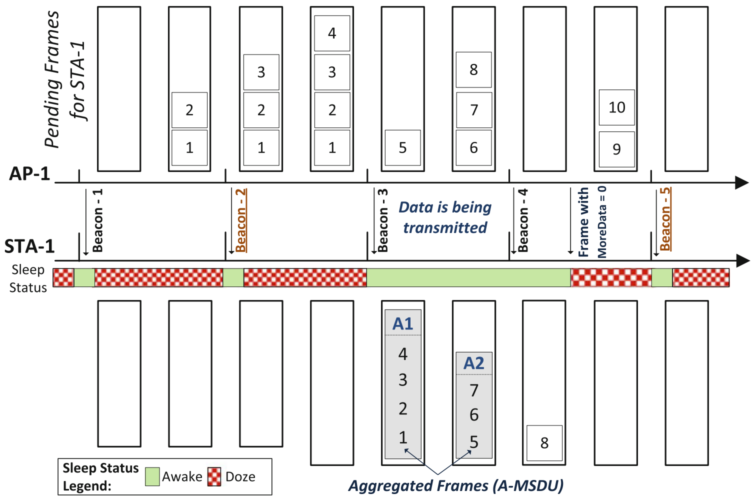


Fig. 1. OPAMA algorithm simplified operation example.

to be sent. As a result, *Frame-A2* carries fewer frames than *Frame-A1*. *Frame-8* was sent without aggregation, since there is only a single frame to be sent.

The end-user feedback will be transmitted to the access point using two distinct messages, *PS-Poll* and *NullFunction*. The first message is used to request data from the AP, while the latter is an empty message used to inform the AP about shifts between two distinct power modes (e.g., going to sleep). Therefore, these message types are only transmitted from the STA to the AP and they do not carry payload data. OPAMA will add one extra byte field to these messages, allowing the STAs to inform the AP about two different performance parameters: the average delay for the last received frames and the maximum allowed delay.

The decision to determine whether or not pending data information should be sent is performed by the OPAMA core algorithm, defined in Algorithm 1. First of all, OPAMA gets all the reference values needed to execute the algorithm, such as the maximum delay allowed by the STA or the aggregation limit support. Later, OPAMA analyzes the pending frames for the current STA, starting by verifying the delay related constraints (*lines 11 to 18*). When analyzing each frame, OPAMA also updates the total pending bytes to be sent (*line 24*) and performs an application dependent assessment (*lines 19-23*). Actually, OPAMA provides specific mechanisms for video applications, where the main goal is to ensure that no more than a pre-defined number of video key frames (α parameter in *line 20*) will be queued. The video key frames parameter is specific to video applications, but all the others mechanisms can be used with mixed traffic scenarios. The performance when handling combined application scenarios might depend on end-user preferences. For instance, the STA maximum allowed delay, can be defined using an algorithm designed to select the best parameter according to the end-user high level preferences for each

Algorithm 1 Determine whether or not pending data information should be sent to a certain STA

```
1: function SEND_PENDING_DATA_TO_STA_DECISION($STA_{MacAddress}$)
2:     ▷ Update the STA maximum allowed delay with information received from the STA in a previous PS-Poll or NullFunction message.
3:     $refresh\_STA\_Maximum\_Allow\_Delay(STA_{MacAddress})$
4:     $STA_{MaxDelay} \leftarrow STAList[STA_{MacAddress}].maxDelay$
5:     $Aggregation_{Threshold} \leftarrow getAggregationThreshold(STA_{MacAddress})$
6:     $TimeUBeacon \leftarrow getTimeUntilNextBeacon()$ ▷ Gets the time until sending next beacon
7:     ...
8:     $PendingFramesList \leftarrow getPendingFrames(STA_{MacAddress})$
9:     $TempPendingBytes \leftarrow NULL$
10:    for each $PFrame$ in $PendingFramesList$ do
11:        ▷ Check if the actual frame delay is greater or equal than the maximum delay defined by the STA
12:        if $getActualDelay(PFrame) > STA_{MaxDelay}$ then
13:            return TRUE
14:        end if
15:        ▷ Check if the sum of actual frame delay with the time until next beacon is greater or equal than the $STA_{MaxDelay}$
16:        if $(getActualDelay(PFrame) + TimeUBeacon) \geq STA_{MaxDelay}$ then
17:            return TRUE
18:        end if
19:        if $PFrame_{MediaType} ==$ "$video$" and $PFrame_{FrameType} ==$ "$I$" then
20:            if $get\_Total\_Video\_KeyFrames\_Pending\_To\_STA(STA_{MacAddress}) > \alpha$ then
21:                return TRUE
22:            end if
23:        end if
24:        $TempPendingBytes \leftarrow TempPendingBytes + PFrame_{SizeBytes}$
25:    end for
26:    if $(TempPendingBytes/Aggregation_{Threshold}) \geq \beta$ then
27:        return TRUE
28:    end if
29:    return FALSE                              ▷ Pending data information will not be sent
30: end function
```

application type. Additionally, the algorithm analyzes the maximum allowed number of aggregated frames to be sent using the STA aggregation limit information ($Aggregation_{Threshold}$) and the total size of current pending data. The parameter β (*line 26*) controls the maximum number of aggregated frames, which can be queued to a certain STA. The configuration of this parameter might also be performed using dynamic approaches where, for instance, the network conditions or frames queuing time in lower layers (e.g. physical) are considered. The aggregation threshold information is associated with each STA (*lines 3-5*), since the maximum feasible aggregation size is related to the STA Maximum Transmission Unit (MTU).

The following section presents detailed information concerning OPAMA performance when compared with the Legacy-PSM and when no PSM is used.

4 Performance Evaluation

This section shows the OPAMA evaluation performed in OMNeT++. First, the simulation details and configuration parameters are given, followed by OPAMA detailed performance analysis. The analysis includes OPAMA performance against Legacy-PSM and no PSM case, and a study concerning OPAMA key configurable parameters.

4.1 Simulation Scenario and Setup

The assessment of OPAMA was performed with two objectives. First, it aims to evaluate the impact of OPAMA on both energy consumption and delay, when compared to Legacy-PSM and no PSM scenarios. Second, to assess how the aggregation threshold influences the behavior of OPAMA.

The tests were conducted in the OMNeT++ 4.2.2 [16] simulator together with the INET Framework 2.0.0. As one of the main goals of this work is to study energy consumption in the IEEE 802.11 interfaces, a multimeter like module, based on the existing INET Framework battery model, was created. This module can measure energy consumed in a IEEE 802.11 interface, by computing the time spent in each state. The simulation scenario used is illustrated in Fig. 2.

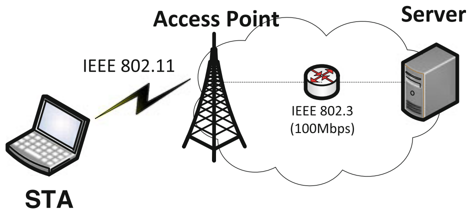


Fig. 2. OMNeT++ IEEE 802.11 simulation scenario

Table 1 illustrates the power values [5] used for each considered state in the IEEE 802.11 physical layer implementation and the key parameters defined for the simulation. Both Legacy-PSM and OPAMA were implemented using the OMNeT++ INET framework. The IEEE 802.11 radio Bit Error Rate (BER) used in this simulation study results from values obtained for various IEEE 802.11g physical modes, using a dedicated Orthogonal Frequency-Division Multiplexing (OFDM) physical layer simulator. The OPAMA related configuration is performed by defining both α and β parameters, used in Algorithm 1.

The assessment of OPAMA was performed using publicly available real traces from a video application [17]. The selected video was the "Sony Demo". This sequence was encoded with MPEG-4 using a Variable Bit Rate (VBR), and has a resolution of 352×288, containing 17000 frames. The video is played for 10 minutes. Additionally, three distinct video qualities were selected for the tests, as summarized in Table 2.

Table 1. OMNeT++ simulation parameters.

Parameter	**Value**
Total simulation time	660 seconds
Number of Runs	20
IEEE 802.11 - Operation mode	G
IEEE 802.11 - Beacon interval	100ms
IEEE 802.11 - Aggregation type	A-MSDU
Radio - Attenuation threshold	-110dBm
Radio - Maximum sending power	2.0mW
Radio - SNIR threshold	4dB
Radio - BER table	"per_table_80211g_Trivellato.dat"
Power while transmitting	2000mW
Power while receiving	1500mW
Power while idle	300mW
Power while sleep	20mW
OPAMA α parameter	10
OPAMA β parameter	3

Table 2. Video traces details.

Name	**Quantizer**	**Mean Frame Bitrate**	**Peak Frame Bitrate**	**Mean Frame Size**
Video-Q1	20	199.91 kbps	2410.56 kbps	832.99 bytes
Video-Q2	12	319.55 kbps	4139.04 kbps	1331.45 bytes
Video-Q3	04	1164.22 kbps	10989.84 kbps	4850.90 bytes

All the results presented in the following sections include 20 runs using distinct random seed numbers with a confidence interval of 95%.

4.2 Results

This subsection presents the attained results regarding OPAMA performance assessment, compared with Legacy-PSM and no PSM scenarios.

OPAMA with No End-user Feedback (OPAMA-NEF): In order to compare the proposed algorithm base implementation against both Legacy-PSM and no PSM scenarios, in this study OPAMA was used without considering the STA maximum allowed delay information (OPAMA-NEF). Nevertheless, OPAMA-NEF still uses aggregation to send multiple packets arriving within a small interval ($\leq$ 5 ms). The maximum aggregation size was defined as 2272 bytes, which is the IEEE 802.11g MTU. This configuration will allow a proper validation against the Legacy-PSM.

Figure 3 depicts a *boxplot* representing the end-to-end delay (in milliseconds) obtained for all the packets needed to stream each of the three distinct VBR videos already presented (Table 2).

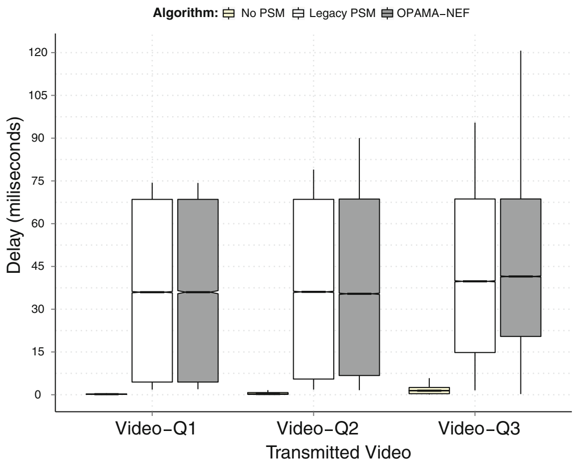


Fig. 3. No PSM, Legacy PSM and OPAMA-NEF end-to-end delay.

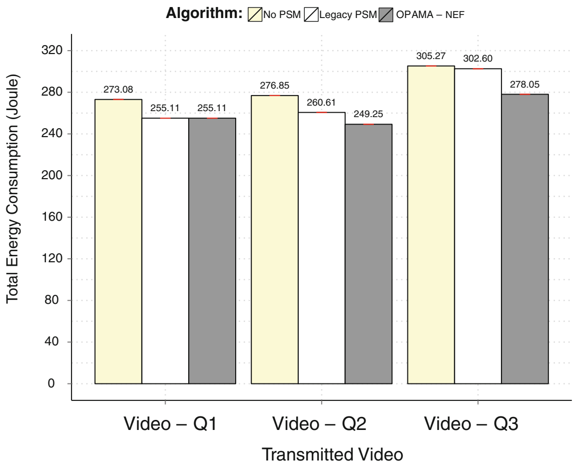


Fig. 4. No PSM, Legacy PSM and OPAMA-NEF energy consumption.

As expected, the scenario where no PSM is used shows a lower delay compared with both Legacy-PSM and OPAMA-NEF. When assessing Legacy-PSM and OPAMA-NEF performance it is noticeable that the delay is similar in both cases. The total energy consumed (in Joule) during the video transmission is illustrated in Fig. 4.

The confidence interval limits are represented by the lines in the top of each bar. Although both Legacy-PSM and OPAMA-NEF introduce extra delay, the energy saving is not significant (only 6% to 8%), which does not configure a good tradeoff between the extra delay introduced and the energy consumed. As discussed previously, this behavior is mainly caused by the limitations of

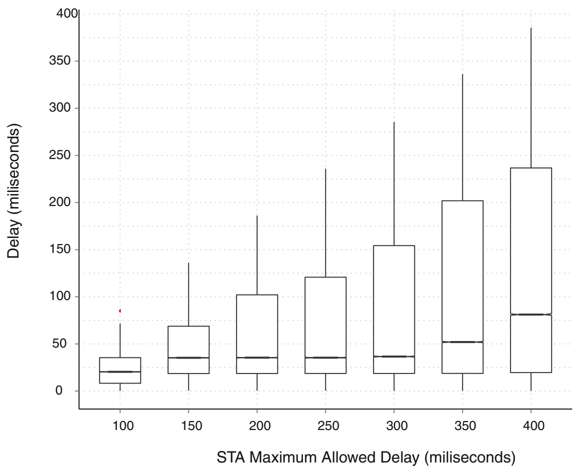


Fig. 5. End-to-end delay for OPAMA with Maximum Allowed Delay defined by the STA.

these power save mode protocols in the case of continuous media applications. Since those applications have almost always data pending to be transmitted, the possibilities for the STA to sleep are very limited. It must be highlighted that unlike OPAMA-NEF, OPAMA will be able to control whether or not the pending data information should be broadcasted to the STA, allowing a better sleep period optimization.

Impact of STA Maximum Allowed Delay on OPAMA performance: This subsection studies the impact of the maximum allowed delay defined by the STA on OPAMA performance. From now on, as the obtained results with the three distinct videos (*Video-Q1*, *Video-Q2* and *Video-Q3*) are similar, and due to lack of space, only Video-Q2 will be used in the analysis. Figure 5 depicts a *boxplot* with the end-to-end delay (in milliseconds) in the y-axis. The x-axis represents the STA maximum allowed delay (in milliseconds). To allow a proper performance comparison, the maximum allowed delay defined by the STA was always kept constant in each test set.

The STA maximum allowed delay was never exceeded for all the test cases. By observing the *boxplots* mean values, it is possible to conclude that end-to-end delay is below 100 ms in all the tested scenarios. This behavior can be explained by the strict delay control performed in conjunction with frame aggregation, as OPAMA always tries to maximize the number of frames sent in each A-MSDU frame. The first quartile analysis also shows that for 25% of the packets, the delay is roughly the same as in the no PSM scenario (see Fig. 3). This fact is directly related with the proper aggregation opportunities created by OPAMA. Additionally, it is also possible to observe that 75% (third quartile) of

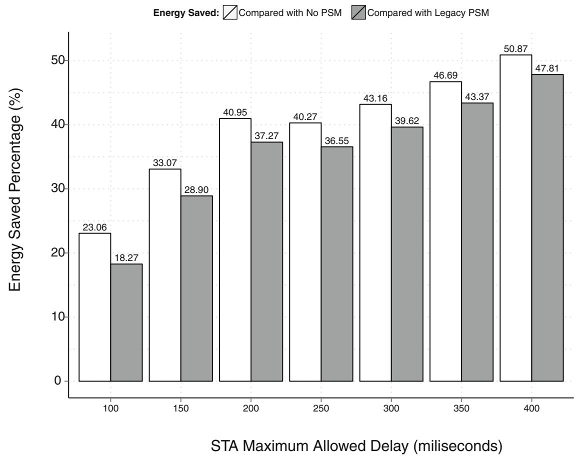


Fig. 6. Energy savings with OPAMA, compared with Legacy-PSM and no PSM scenarios.

the delivered packets have only around a half of the maximum delay tolerated by the STA.

A comparison of the obtained energy savings regarding the employment of OPAMA compared with both Legacy-PSM and no PSM scenario is shown in Fig. 6. The y-axis represents the energy saved in percentage, while the maximum allowed delay defined by the STA is depicted in the x-axis.

The results show benefits of using OPAMA, when the STA can accommodate some delay (e.g., by using a local buffering technique). The savings for the 100 ms maximum delay when compared with the Legacy-PSM are around 18%, which in this particular case allows the end-user to play the video for almost two minutes more using the same energy. Higher maximum allowed delays, such as 300 ms, boost the savings to around 40%. At a first glance, it might not seem interesting to employ such large delays. However, the STA can dynamically inform the OPAMA ready AP about the maximum expected delay to reflect the end-user behavior.

OPAMA performance with larger aggregation threshold: OPAMA uses MAC layer aggregation (A-MSDU) as one of the algorithm components. Until now, all the tests were performed using a maximum aggregation size of 2272 bytes, which is the MTU for IEEE 802.11g. Nevertheless, in IEEE 802.11n, where aggregation at the receiver side is already mandatory, the MTU can be up to 7935 bytes. Therefore, this section investigates the OPAMA behavior with two distinct maximum aggregation sizes.

Figure 7 shows the end-to-end delay for aggregation threshold size of 2272 and 7935 bytes. The x-axis represents the maximum allowed delay by the STA.

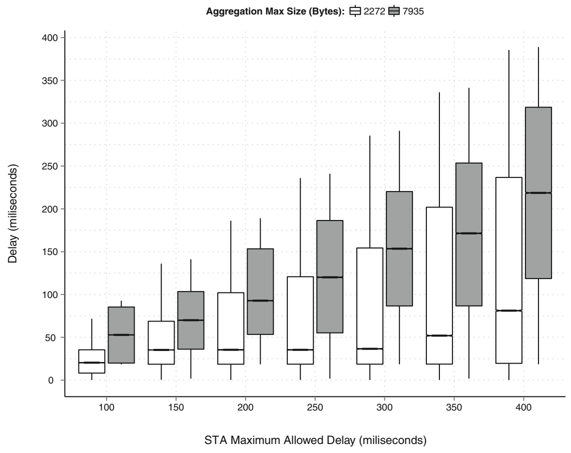


Fig. 7. Aggregation threshold impact on OPAMA delay.

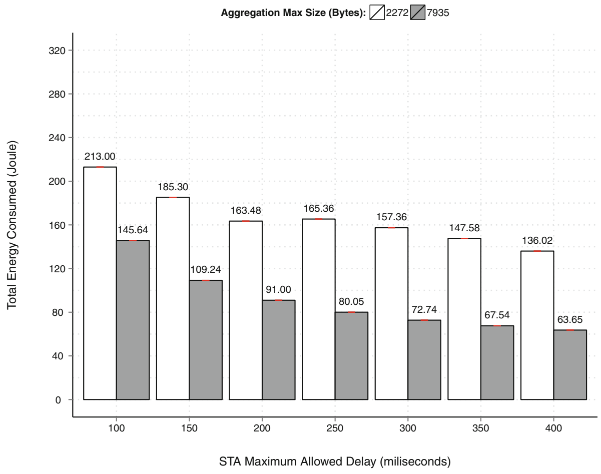


Fig. 8. Aggregation threshold impact on OPAMA energy consumption.

For both scenarios, the maximum delay is not greater than the one defined by the STA. When comparing the two aggregation scenarios, the one with the lowest aggregation size shows a lower mean delay. If only the end-to-end delay is analyzed this might reveal poor OPAMA performance with larger aggregation threshold. However, the information regarding energy consumption, shown in Fig. 8, highlights the opposite.

When employing larger aggregation threshold (i.e., 7935 bytes) OPAMA introduces additional delay for almost all the delivered packets. The energy consumption is significantly lower. Therefore, OPAMA clearly improves the cost/benefit tradeoff between delay and energy consumption under these conditions. The usage of larger aggregation frames also reduces the number of MAC layer acknowledgments in the network, which reduces the global network contention and maximizes the STA sleep time.

The usage of 7935 bytes as maximum aggregation threshold, when STA maximum allowed delay is defined as 100 ms, is able to achieve savings of 32%. Moreover, when comparing OPAMA performance under these conditions with Legacy-PSM, the savings are up 44%. The savings for a STA allowing a maximum delay of 300 ms and 400 ms are 74% and 76%, respectively.

5 Conclusions

The energy efficiency in the end-user IEEE 802.11 ready devices is still an important factor towards a fast and global deployment of the future Internet of Things paradigm, since battery lifetime is one of the most critical factors in a daily usage. This paper investigates and proposes a mechanism aiming at saving energy while supporting continuous media applications. The proposed power save algorithm for IEEE 802.11 networks, named OPAMA, was designed to enhance the energy consumption by extending the IEEE 802.11 legacy PSM in order to accommodate the end-user feedback, and using Aggregated MAC Service Data Unit (A-MSDU) to deliver data frames.

OPAMA performance assessment showed capabilities to improve energy efficiency, while keeping the end-user expectation at the defined level. When compared with the IEEE 802.11 Legacy-PSM, the OPAMA proposal achieved energy savings up to 76% in a higher tolerable delay scenario and 44% for a scenario where the STA can only accommodate a maximum delay of 100 ms. The impact of the aggregation threshold in the proposed algorithm performance was also noticeable, depicting considerable energy savings.

Acknowledgments. This work was partially supported by the iCIS project (CENTRO-07-ST24-FEDER-002003), co-financed by QREN, in the scope of the Mais Centro Program and European Union's FEDER, as well as by the COST framework, under Actions IC0804 and IC0906. The first author was also supported by the Portuguese National Foundation for Science and Technology (FCT) through a Doctoral Grant (SFRH/BD/66181/2009).

References

1. Tozlu, S., Senel, M., Mao, W., Keshavarzian, A.: Wi-fi enabled sensors for internet of things: a practical approach. IEEE Commun. Mag. **50**(6), 134–143 (2012)
2. IEEE: ANSI/IEEE Std 802.11, 1999 Edition (r2003), i -513 (2003)
3. IEEE: IEEE std 802.11e-2005 (amendment to IEEE std 802.11-1999), 0–189 (2005)
4. IEEE: IEEE std 802.11n-2009 (amendment to IEEE std 802.11-2007), 1–565 (2009)
5. Camps-Mur, D., Gomony, M.D., PéRez-Costa, X., Sallent-Ribes, S.: Leveraging 802.11n frame aggregation to enhance qos and power consumption in wi-fi networks. Comput. Netw. **56**(12), 2896–2911 (2012)

6. Tan, E., Guo, L., Chen, S., Zhang, X.: Psm-throttling: minimizing energy consumption for bulk data communications in wlans. In: IEEE International Conference on Network Protocols, ICNP 2007, pp. 123–132, October 2007
7. Adams, J., Muntean, G.M.: Adaptive-buffer power save mechanism for mobile multimedia streaming. In: IEEE International Conference on Communications, ICC '07, pp. 4548–4553, June 2007
8. Palit, R., Naik, K., Singh, A.: Impact of packet aggregation on energy consumption in smartphones. In: 2011 7th, International Wireless Communications and Mobile Computing Conference (IWCMC), pp. 589–594, July 2011
9. Skordoulis, D., Ni, Q., Chen, H.H., Stephens, A., Liu, C., Jamalipour, A.: IEEE 802.11n mac frame aggregation mechanisms for next-generation high-throughput wlans. IEEE Wirel. Commun. **15**(1), 40–47 (2008)
10. Lorchat, J., Noel, T.: Reducing power consumption in IEEE 802.11 networks. In: IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2005), vol. 2, pp. 106–110, August 2005
11. Kennedy, M., Ksentini, A., Hadjadj-Aoul, Y., Muntean, G.M.: Adaptive energy optimization in multimedia-centric wireless devices: a survey. IEEE Commun. Surv. Tutorials **PP**(99), 1–19 (2012)
12. Pathak, A., Hu, Y.C., Zhang, M.: Where is the energy spent inside my app?: fine grained energy accounting on smartphones with eprof. In: Proceedings of the 7th ACM European Conference on Computer Systems. EuroSys '12, pp. 29–42. ACM, New York (2012)
13. Naik, K.: A survey of software based energy saving methodologies for handheld wireless communication devices. Tech. Report No. 2010–13, Dept. of ECE, University of Waterloo (2010)
14. Dogar, F.R., Steenkiste, P., Papagiannaki, K.: Catnap: exploiting high bandwidth wireless interfaces to save energy for mobile devices. In: Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services. MobiSys '10, pp. 107–122. ACM, New York (2010)
15. Zhu, Y., Lu, H., Leung, V.: Access point buffer management for power saving in IEEE 802.11 wlans. IEEE Trans. Netw. Service Manag. **9**(4), 473–486 (2012)
16. Varga, A., Hornig, R.: An overview of the omnet++ simulation environment. In: Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops. Simutools '08, ICST, Brussels, Belgium, Belgium, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), pp. 60:1–60:10 (2008)
17. Van der Auwera, G., David, P., Reisslein, M.: Traffic and quality characterization of single-layer video streams encoded with the h.264/mpeg-4 advanced video coding standard and scalable video coding extension. IEEE Trans. Broadcast. **54**(3), 698–718 (2008)

Real-World Energy Measurements of a Wireless Mesh Network

A. Jamakovic[1(✉)], D.C. Dimitrova[1], M. Anwander[1], T. Macicas[1], T. Braun[1], J. Schwanbeck[2], T. Staub[1], and B. Nyffenegger[1]

[1] Communication and Distributed Systems Group, Institute of Computer Science and Applied Mathematics, University of Bern, Bern, Switzerland
{jamakovic, dimitrova, anwander, macicas, braun, staub, nyffenegger}@iam.unibe.ch

[2] Hydrology Group, Oeschger Centre for Climate Change Research and Institute of Geography, University of Bern, Bern, Switzerland
jan.schwanbeck@guib.unibe.ch

Abstract. Over the past several years the topics of energy consumption and energy harvesting have gained significant importance as a means for improved operation of wireless sensor and mesh networks. Energy-awareness of operation is especially relevant for application scenarios from the domain of environmental monitoring in hard to access areas. In this work we reflect upon our experiences with a real-world deployment of a wireless mesh network. In particular, a comprehensive study on energy measurements collected over several weeks during the summer and the winter period in a network deployment in the Swiss Alps is presented. Energy performance is monitored and analysed for three system components, namely, mesh node, battery and solar panel module. Our findings cover a number of aspects of energy consumption, including the amount of load consumed by a mesh node, the amount of load harvested by a solar panel module, and the dependencies between these two. With our work we aim to shed some light on energy-aware network operation and to help both users and developers in the planning and deployment of a new wireless (mesh) network for environmental research.

Keywords: Wireless mesh networks · Real-world deployments · Energy measurements · Harvesting and consumption · Energy efficiency

1 Introduction

Technical advances in recent decades have led to availability of a wide range of reasonably priced sensors for hydro-meteorological, or generally any other type of systems for environment monitoring. Still, the wide spread-deployment of such systems depends on the costs for data-transfer and maintenance, especially for sensor networks placed in remote or hardly accessible sites. To tackle the connectivity issue, current research combines (wireless) sensor networks with wireless mesh networks to obtain a more optimized system solution. In particular, a wireless mesh network is used to connect separate sensor networks with

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 218–232, 2013.
DOI: 10.1007/978-3-642-40517-4_18,

e.g. a monitoring platform, or as a scalable backbone for sensor-to-sensor communication by connecting the separate sensor networks associated to each mesh node.

As more and more outdoor applications require long-lasting, highly energy-efficient, and in fact continuously-working mesh networks running on battery-powered mesh nodes, research on multiple aspects of energy consumption becomes critical. One often taken path by researchers is developing solutions to decrease the energy cost of operation of, e.g., energy-aware routing protocol, algorithms to improve communication schemes, etc. While we acknowledge the importance of this research, we argue that it is of utmost importance to study the problems of energy provisioning as well. Running on batteries is an excellent solution for node independence but unfortunately faces the issue of possible battery depletion. Leaving aside convenience, change of the battery is not feasible for difficult to access deployments, as the one we address in this paper. Therefore, the problem of energy harvesting and energy consumption plays an increasingly important role.

The choice of an appropriate energy harvesting strategy mainly depends on the expected lifetime of the deployed network, the actual energy consumption of the nodes and the physical requirements towards the node design. Although there are several studies that consider energy consumption by wireless sensor or mesh nodes, to the best of our knowledge, none of them conducts investigations in on-site deployed network. In this paper we aim to bridge the gap by offering analysis of a large amount of energy-related measurement data, collected in a real-world wireless mesh network.

The real-world wireless mesh network, referred to as A^4-Mesh[1], was designed and deployed in the Swiss Alps to serve the needs of distant hydro-meteorological monitoring in remote locations. Its purpose is to provide researchers with secure access to their measurements of interest, irrespective of the location. To this extent, the mesh network should support communication to the sensing devices at all times, irrespectively of whether to collect measurement data or to send control commands. To ensure constant availability of the mesh backbone we rely on solar energy.

Our contribution in this paper focuses on providing feedback on the energy-related operation of a wireless mesh network in a real-world deployment. Both energy consumption and energy provisioning are considered. In particular, we conduct measurements to monitor and analyse the energy behaviour of the wireless mesh network from three different perspectives, namely, the mesh node, the battery and the renewable energy source (solar panel module in our case). We believe that our findings can provide an important basis to help researchers and other interested parties in the planning and deployment phase of a wireless mesh network supporting environmental monitoring.

[1] A^4-Mesh project and its currently ongoing extension aims at developing a completely functional wireless mesh infrastructure including support for authentication and authorization, accounting, and auditing. For more information, please visit project's website https://a4-mesh.unibe.ch/

The remainder of the paper is organized as follows. Section 2 gives an overview of recent work regarding the deployment of wireless mesh networks, hardware and software approaches to tackle the problem of energy efficiency. Section 3 discusses the design and realisation of the wireless mesh network. Section 4 presents the energy monitoring setup and our findings on energy harvesting and energy consumption in the deployed wireless mesh network. Finally, Sect. 5 summarizes the paper and outlines several interesting problems for further research.

2 Related Work

Wireless mesh networks (WMNs) have been subject to intensive research for several years. This is mainly due to the fact that WMNs imply both the ad-hoc and the traditional infrastructure model of access networks, offering more flexibility but also posing various new challenges for researchers and developers. A comprehensive review on the main research challenges in wireless mesh networks is provided in [1,2], and the most recent one in [3].

Generally, a WMN consists of wireless mesh nodes, which can operate as hosts but also as routers, being in this way access point for the mesh clients. The mesh nodes are typically fully functional computers with tailored embedded operating systems so as to match hardware resource constraints. The mesh clients are often laptops, cell phones or other wireless devices, through which various measurement devices such as sensors can be connected. Single-hop as well as multi-hop modus operandi is supported. The flexibility of mesh nodes in forming a network makes them appropriate choice to form the communication backbone for a large range of application scenarios, including environmental and habitat monitoring, safety surveillance and many others.

First outdoor applications of WNMs have been reported in e.g. [4] for serving local flood warning system and in [5] for facilitating sensor data collection for ecological researchers in a natural reserve. Other applications concern applying wireless mesh networks for safety surveillance in industry or public environments, tactical support for the military, as well as many public and commercial service networking scenarios. An extensive overview of these can be found in [1]. Although each of these scenarios poses various research challenges in terms of routing, connectivity or data processing, in this paper we are more interested in the aspect of energy-operation.

Energy-aware design for wireless mesh networks has been studied from many angles. In the context of the project SolarMESH [6], the potential of powering 802.11-based mesh networks using solar power and rechargeable batteries has been examined. In particular, the problem of lower power operation at the network layer is addressed by the authors. Then, from the perspective of provisioning each node with a solar panel and battery combination that is sufficient to prevent node outage for the duration of the deployment, one could refer to [7,8]. Related to it, the resource allocation and outage control for solar-powered WLAN mesh networks is considered in [9].

Furthermore, in [10] authors propose a context-aware energy management system for network nodes that are energy-self-sufficient. Moreover, a battery-aware scheme for energy efficient coverage and routing is proposed in [11]. In addition an energy model for network coding-enabled WMN, based on the IEEE 802.11 technology, is studied in [12]. Lastly, in [13] authors investigate the energy consumption behavior, though from a perspective of a wireless network interface in an ad hoc networking environment. However, to the best of our knowledge, no study attempts to determine the energy generation and the energy consumption of a real-world wireless mesh network, nor it, in that way, focuses on the energy-aware approach to the deployment of a new wireless (mesh) network. Our paper makes this attempt by measuring and analysing a large amount of energy-related measurement data of a real-world wireless mesh network.

3 WMN Supporting Environmental Research

Deployment of a real-world wireless network requires careful design and meticulous consideration of various aspects related to its way of operation, communication, tooling, etc. Failing to do so can lead to poor or even erroneous network performance. Accordingly, the deployment of the A^4-Mesh network has proceeded in steps, considering system purpose, deployment environment, and technology solutions.

3.1 WMN Deployment Scenario

The application scenario, supported by the A^4-Mesh network, considers an hydro-meteorological monitoring network in the field that should be remotely accessible by researchers at university laboratories. The monitoring network consists of different measurement devices (i.e., sensors) located at various remote sites. From the researchers' perspective it is very desirable and highly convenient to access the devices directly from the campus site, ensuring in this way data transfer at frequent intervals as well as the possibility of remote control, which both reduce the risk of data loss. Hence, a large amount of data, produced by the sensors, needs to be transferred to the university campus, preferably in near real-time. Moreover, a control channel to the sensor network should be supported as well.

In the A^4-Mesh network we propose to use a wireless mesh network as the communication backbone. It is a common form of WMN, where each mesh node relays data for other mesh nodes (a typical adhoc networking paradigm) and only few mesh nodes can additionally act as routers, becoming Internet gateways. The wireless mesh nodes interconnect their corresponding hydro-meteorological sensor(s) to the university campus network through the wireless mesh gateway. Hence, the deployed A^4-Mesh network successfully serves the purpose of a communication infrastructure for environmental monitoring, giving researchers low-cost broadband network access. Figure 1 shows the network setup with the distance of each wireless link and the locations of the connected environmental monitoring stations.

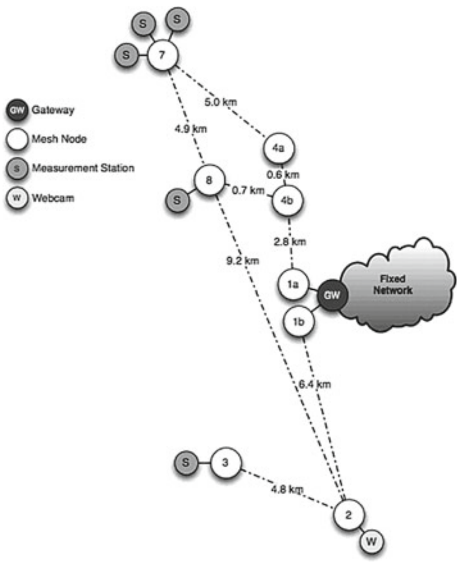


Fig. 1. Wireless mesh network deployed for support to environmental research.

The A^4-Mesh network is deployed in the Swiss Alps, in the Valais region. Figure 1 shows the network setup, used in our measurements, with the distance of each wireless link and the locations of the connected environmental monitoring stations. Node 1 serves as a gateway to the fixed network and thus carries the traffic between the mesh nodes and the Internet. The first link from node 1 is directed to a relay station (node 2) in Vercorin at the opposite hill slope, where a webcam is located. The second link from node 1 connects to nodes 4a and 4b in Cry d'Er. Node 3 connects to the gateway through node 2. Clients, i.e. measurements stations, connect to the mesh nodes using wireless or wired links.

The region where the wireless mesh network is deployed (in the Valais region of the Swiss Alps) has extreme weather conditions with a lot of snow in the winter, requiring mesh nodes that are specifically built for these conditions, i.e. durable cases, self-contained power supply, etc. The latter is especially important and should be able to provide energy to the node at any time. We chose for a solar panel module accompanied by batteries powerful enough to bridge periods of poor sunlight during the long winter period. The panels should be at least 3 meters above the ground to prevent them from being covered by snow. The initial and the final installation of one of the wireless mesh nodes are shown in Fig. 2.

In order to allow for easy access to the A^4-Mesh network, special care was taken regarding the seamless integration of authorisation and authentication functionality into the organization's own authentication and authorisation infrastructure (AAI). In our particular case an existing AAI Shibboleth-based federation, namely, SWITCHaai was used [14]. SWITCHaai is operated by SWITCH,

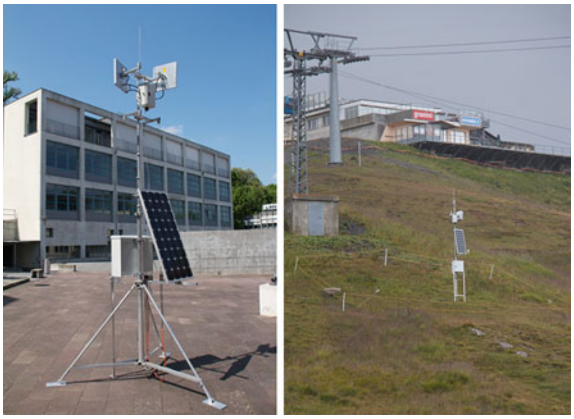


Fig. 2. Initial and final installation setup of a wireless mesh node for support to environmental research.

the Swiss National Research and Education Network operator, and offers convenient access to academic resources through a single ‘virtual ID’.

3.2 WMN Design and Realisation

In current wireless networks, the design of wireless mesh nodes differentiates between two main building blocks, namely, the computer and the wireless hardware. The computer is composed of all those components that are generic to any networking platform and provide the main processing operations required to support the networking process. On the contrary, the wireless hardware refers to the components of the node specific to the wireless transmission process. This includes the wireless cards and other RF components that might be used on the node. Figure 3 shows a block diagram of our wireless mesh node design. It consists of two main system boards, on the left and right side, and one additional backup board in the middle with support for cellular connectivity. The specific

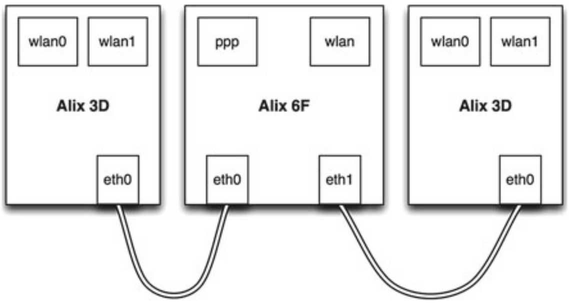


Fig. 3. A block diagram of the wireless mesh node design.

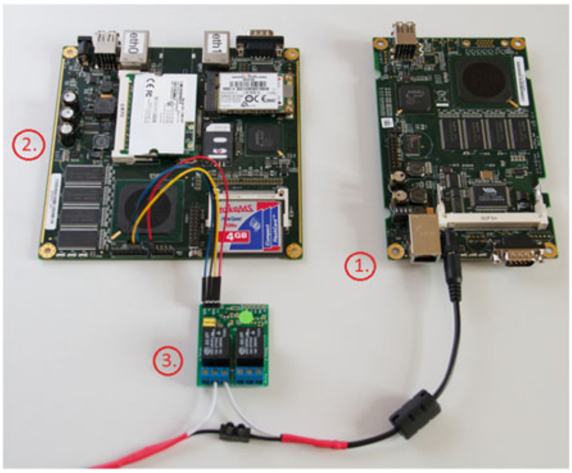


Fig. 4. An example of the main board (1) together with the backup board (2) and the controlling reboot relay (3) of a wireless mesh node used in the deployment of the A^4-Mesh network.

design choice is motivated by the need for remote recovery in case of failure of the mesh node. If an erroneous image is uploaded to a mesh node, we are still able to connect to the backup board through the cellular networks and either reboot the node or upload a new image. Hence, the need to physically access a temporally non-operational node is omitted, reducing maintenance costs of the overall WMN. The use of a backup board additionally improves the node uninterrupted operational lifetime, or at least for nodes covered by cellular networks.

For the realisation of the mesh nodes we selected PC Engines ALIX boards, which are small form factor system boards optimized for wireless routing and network security applications. ALIX.3 series boards are used as the two main system boards while an ALIX.6 series board with support for GSM/UMTS cards is used for the backup board. Figure 4 shows an example of interconnected main system and backup board of a mesh node we used in our deployment. Then, in terms of wireless hardware, the two ALIX.3-series boards have 802.11n radio cards installed in the miniPCI socket. These are used to provide the interconnection link between the mesh nodes. Note that on each main system two cards are installed, supporting in total up to four links per one mesh node. The ALIX.6-series backup board has an 802.11 a/b/g radio card installed in the miniPCI socket, providing wireless connectivity, and an UMTS embedded module of Sierra Wireless AirPrime MC Series.

As the current deployment of the A^4-Mesh wireless mesh network is based on the IEEE 802.11n standard, appropriate antennas are necessary. 802.11n builds on previous 802.11 standards by adding multiple input multiple output (MIMO) technology to improve network throughput. Therefore, we used MIMO 25 dBi Dual Polarization Panel antennas, which are designed for the 5.8 GHz frequency

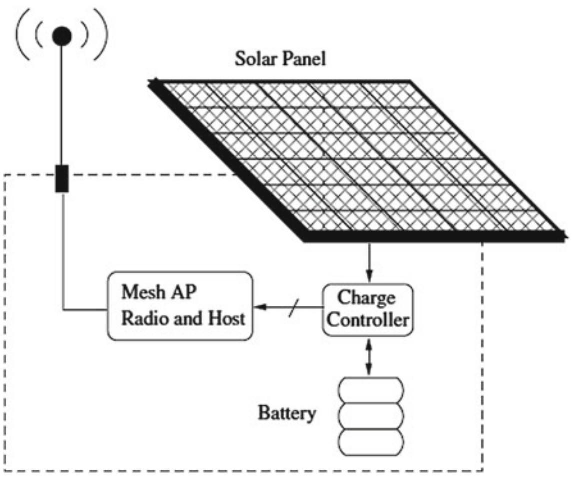


Fig. 5. Schematic of a simple solar panel charger circuit, taken from [7].

range and, at rating of 25 dBi, perfectly suited for our long distance point-to-point communication. Furthermore, the mesh node is connected to a twin relay that works with the standard I2C interface to control and individually reboot up to two connected mesh boards easily and without taking up any ports or MPU resources. Relays are configured to conduct electric current to the connected board when turned off.

The whole operation of the mesh nodes is supported by a solar-based power system. Solar energy is harvested by means of a single solar panel per node, i.e., the STM 90 model by SunTechnics with nominal voltage of 18.75 V. The panel is especially appropriate for off-grid solar installations due to its light weight and convenient dimensions. The panel is used to feed the mesh node or to charge the battery. We are using the NPL78-12FR 78 Ah model by Yuasa, which is a sealed lead acid battery. In order to maximise the battery charging state and protect from over-discharge a Maximum Power Point Tracking (MPPT) controller is included, in particular the MorningStar's Sun Saver SS-MPPT-15L model is used. The charging circuit is depicted in Fig. 5.

4 Energy Measurements

Energy measurements were taken over two periods of several weeks each, one in summer and one in winter. In summer, measurements were taken for node 7 and node 8, and in winter for node 3 and node 7 (node 8 had to be replaced in winter for safety of the ski pistes near which it was initially located). Data was collected on the voltage (in Volts) of the solar panel and the battery as well as on the load (in Ah) of the mesh node and the battery. All measurements were taken by programming the SunSaver MPPT charge controller to log measurements data at specified intervals throughout the day and night.

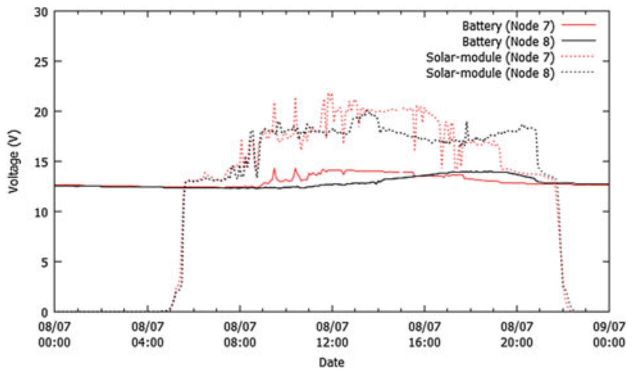


Fig. 6. Measurements of voltage at the battery and the solar panel module of two mesh nodes for an arbitrary summer day taken randomly from the data set.

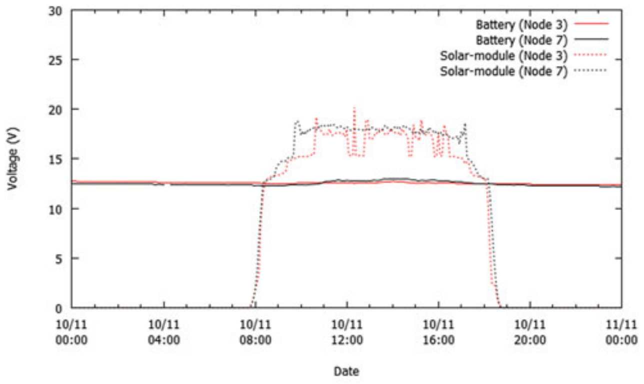


Fig. 7. Measurements of voltage at the battery and the solar panel module of two mesh nodes for an arbitrary winter day taken randomly from the data set.

We begin the discussion with analysing the behaviour of the voltage of both the solar panel and the battery over a single day, taken arbitrary from the data set. The graphs in Figs. 6 and 7 show the results for the summer and winter period, respectively. Although there is variation in the solar panel voltage, significant at times, the average voltage remains high above 15 V in both summer and winter periods, implicating that sufficient node provisioning and battery charging (a 12 V battery charges at about 14 V) can be achieved. The peak fluctuations in the voltage observed during the daytime are primarily due to shadows or clouds, in which cases the battery seamlessly takes over powering the mesh node. Moreover, we observe that the solar panel voltage is influenced not only by the season (higher voltage is recorded in summer) but also by the node location (the daily voltage variations differ per node).

Some other, straightforward observations for the solar panel are: (1) as it can be expected, as soon as the sun goes down, light stops hitting the solar panel

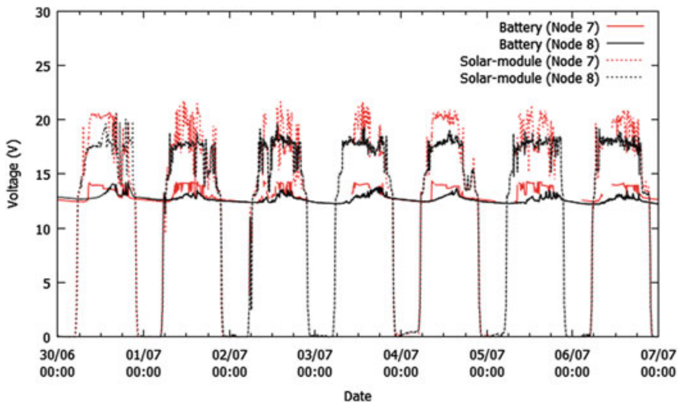


Fig. 8. Measurements of voltage at the battery and the solar panel module of mesh node 8 for one week taken randomly from the data set.

and its voltage drops instantly from a high of above 15 V to a low of 0 V; (2) also intuitive is the longer solar panel activity period in summer due to longed daylight time, e.g., during summer the battery only switches in as a back-up after approximately 9PM until about 7AM (see Fig. 6) while during the winter it does so from about 7PM to 9AM (see Fig. 7). Note that the solar panel generating energy does not necessary mean that the battery is not used at all.

Moving the focus to the battery, its voltage curve appears constant throughout the night when the battery is supplying energy to the mesh node. During daytime there are slight fluctuations in the battery voltage, which is due to the charging of the battery by the solar module. This is especially well visible for the summer period, when one can expect more intensive sunlight. Our observation is further confirmed by looking at a week-long measurement in the summer period as shown in Fig. 8. The figure clearly shows that battery voltage of both nodes can vary considerably over daytime, always above 12 V due to charging, and does so correspondingly due to the variations in the voltage of the solar panel. Moreover, the graphs in Fig. 8 reveal that the voltage values of both the battery and the solar panel module display the same repetitive behaviour during the course of days. During a single day (daytime plus night), the voltage within the mesh node system will be either around 12 V (below which the battery supplies power to the mesh node) or around 18–20 V (when the solar panel module powers the mesh node and recharges the battery). The same measurements but made in winter do not show significant deviations in general behaviour and are omitted here because of space constraints.

The relation between solar panel and battery voltage is well visible in Figs. 9 and 10. The data was collected over a period of few weeks in summer and winter, respectively. We see that the voltage at which the solar panel module produces maximum power in summer is at maximum around 24 V, while in the winter period the voltage reaches a maximum of only around 22 V. At the same time

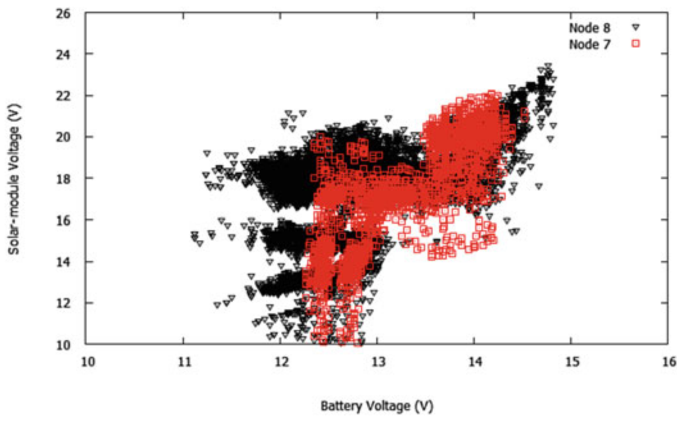


Fig. 9. Relationship between battery and solar panel module voltage. Measurements were taken at two nodes from our real-world wireless mesh network, for a period of few weeks in summer.

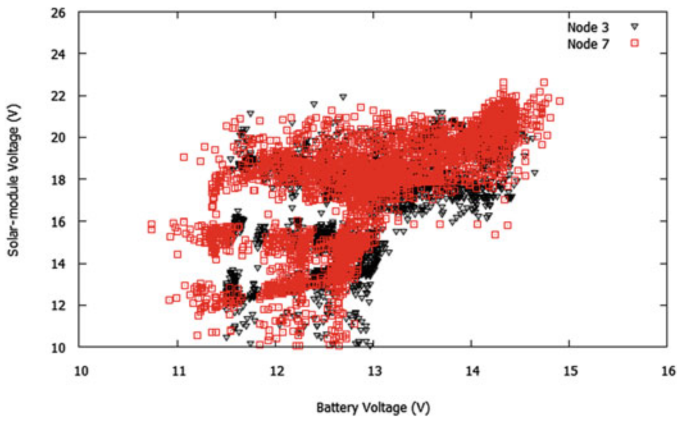


Fig. 10. Relationship between battery and solar panel module voltage. Measurements were taken at two nodes from our real-world wireless mesh network, for a period of few weeks in winter.

lower voltage states are more often reached by the battery voltage in winter, also corresponding to more extensive use of the battery in winter. We also see that as long as the solar panel voltage is above 16 V the battery can charge (its voltage reaches up to 14.5 V), which is in line with the nominal voltage of the panel. It should be noted that delivering charge to the battery at 16 V and delivering enough energy to support the mesh node and battery are not the same. It is possible, as the figures show, that the solar power generates energy but which may not be sufficient to fully recharge the power consumed by the node. Below solar panel voltage of 16 V the battery is mainly discharging with sporadic charges, which are nevertheless not enough to compensate for the discharge. The

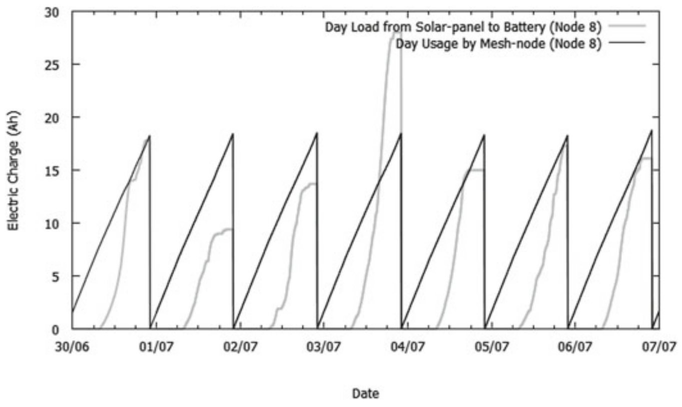


Fig. 11. Battery day load (Ah) along with the mesh node day usage (Ah). Measurements were taken at the mesh node 8 from our real-world wireless mesh network, for a period of one week in summer.

relationship between the battery voltage and the mesh node voltage was also analysed. We observed that they are positively correlated, implying that if one variable increases, the other variable also increases and vice versa. The figures are omitted here because of space constraints.

Along with voltage measurements we also collected data on the electric charge (expressed in Ampere-hours) of the battery and the mesh node. Figure 11 displays the battery day load together with the mesh node day usage for an arbitrary week in summer from the data set. The measurements concern mesh node 8, which is central to the wireless mesh network, see Fig. 1. In addition, Fig. 12 plots the same parameters but for an arbitrary week in the winter period in

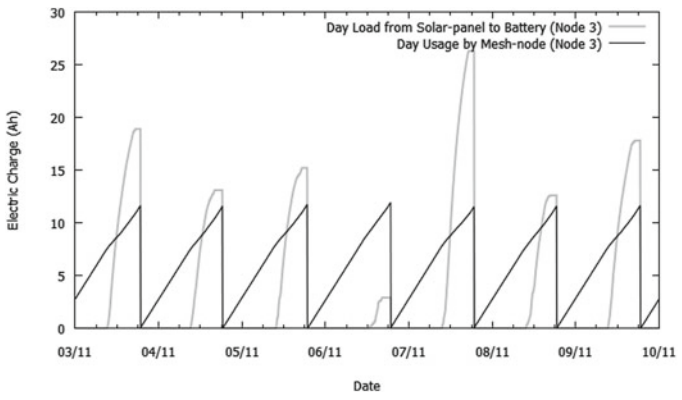


Fig. 12. Battery day load (Ah) along with the mesh node day usage (Ah). Measurements were taken at the mesh node 3 from our real-world wireless mesh network, for a period of one week in winter.

the data set but for the node 3. Monitoring the consumption of node 3 allows us to compare the energy usage between nodes based on their role in the mesh network.

In both figures at first sight it seems that there is lacking data for the battery charge. As the battery day load denotes the battery charging by the solar panel module, we can expect activity only when the panel actually generates charge, which explains the lack of recorded data at the beginning and end of each day when sunlight stops hitting the panel. The case is more extreme in Fig. 12 due to the shorter period of daytime. Moreover, it is interesting to note that the battery day load registers large differences from one day to the next, indicating that, depending on weather conditions, the amount of sunlight reaching the panel may vary considerably. Although sunny days can be used to bring back the battery to full charge and compensate for periods (days) with poor sunlight, one should always consider daylight statistics of potential deployment locations.

Comparing the energy usage of the two mesh nodes shows that node 8 needs more energy (approximately 17 Ah in a day) compared to what node 3 consumes (approximately 12 Ah in a day). The measured data indisputably points out that the mesh node 8 is involved in more intensive inter-node communication using in that way more energy. The impact of the length of the communication link on the required transmit power should also not be neglected, mainly because of higher per-link transmission power. Hence, the design of a wireless mesh network relying on solar energy for its sustainable operation, should take into account the role of each individual node in the overall mesh network as well as the node's location, which affects communication distances but also the amount of usable sunlight.

5 Conclusions and Further Work

In this paper, we have presented a study on energy measurements of a real-world deployment of a wireless mesh network. The measurements were collected over several weeks during the summer and winter period. Although there are several studies that consider various aspects of energy in wireless sensor or mesh nodes, to the best of our knowledge, none of them conducts investigations on energy harvesting and energy consumption in a onsite deployed network. We performed measurements on the mesh nodes but also on the battery and solar panel used to support their operation.

We first analysed the changes in battery and solar panel voltage over a single day or several days. The results revealed that during the span of a single day the voltage either comes at a value of about 14 V, supplied by the battery when the solar panel is not active, or it can reach a maximum voltage value of 24 V (in summer), when the solar panel module powers both the mesh node and recharges the battery. For a period of less sunny weather (in winter) the maximum generated voltage drops to 22 V. Moreover, we observed that during daytime the voltage of the panel may fluctuate, primarily due to shadows or clouds, which results that sometimes during the day the battery seamlessly takes over powering the mesh node. Additionally, we discussed the correlation between the two voltages for a long period of several weeks.

We also presented several results on electric charge from the solar panel module to the battery (i.e. the day load) and the electric charge of the mesh node (i.e. its day usage) during several weeks in both the summer and the winter period. Presented measurements reveal that those nodes that are central (in terms of routing) to the deployed wireless mesh network consume more energy, pointing to a more intensive inter-node communication of that particular mesh node. We hope that our findings can provide an important basis to help researchers and other interested parties in the planning and the development phase of a wireless mesh network supporting environmental monitoring. Besides, we intend to use the current results in the development of a more efficient prediction algorithm for identifying and reducing outages of a solar-power WMN, in that way addressing the power resources in the most efficient and equitable way.

Acknowledgment. We gratefully acknowledge support from: (a) the A^4-Mesh project, carried out as part of the AAA/SWITCH e-Infrastructure for e-Science program which is lead by SWITCH, the Swiss National Research and Education Network, and is supported by funds from the Swiss State Secretariat for Education and Research, (b) the COST (European Cooperation in Science and Technology) framework, under Action IC0804, which supported this work partially, and (c) all persons who made possible the collection of real-world data.

References

1. Akyildiz, I.F., Wang, X.: Wireless mesh networks: a survey. IEEE Commun. Mag. **43**(9), 23–30 (2005)
2. Moustafa, H., Javaid, U., Meddour, D.E., Senouci, S.M.: A panorama of wireless mesh networks: Architecture, application and technical challenges. In: International Workshop on Wireless Mesh: Moving towards Applications, WiMeshNets (2006)
3. Pathak, P.H., Dutta, R.: A Survey of network design problems and joint design approcahes in wireless mesh networks. IEEE Commun. Surv. Tutorials **13**(3), 396–428 (2011)
4. Smith, P.J., et al.: Towards the provision of site specific flood warnings using wireless sensor networks. Meteorol. Appl. **16**(1), 57–64 (2009)
5. Wu, D., Gupta, D., Mohapatra, P.: QuRiNet: a wide-area wireless mesh testbed for research and experimental evaluations. Ad-hoc Netw. **9**, 1221–1237 (2011)
6. Todd, T.D., Sayegh, A.A., Smadi, M.N., Zhao, D.: The need for access point power saving in solar powered WLAN mesh networks. IEEE Network **22**(3), 4–10 (2008)
7. Badawy, G.H., Sayegh, A.A., Todd, T.D.: Solar powered WLAN mesh network provisioning for temporary deployments. In: IEEE Wireless Communications and Networking Conference, WCNC (2008)
8. Badawy, G.H., Sayegh, A.A., Todd, T.D.: Energy provisioning in solar-powered wireless mesh networks. IEEE Trans. Veh. Technol. **59**(8), 3859–3871 (2010)
9. Farbod, A., Todd, T.D.: Resource allocation and outage control for solar-powered WLAN mesh networks. IEEE Trans. Mob. Comput. **6**(8), 960–970 (2007)

10. Gladisch, A., Daher, R., Lehsten P., Tavangarian, D.: Context-aware energy management for energy-self-sufficient network nodes in wireless mesh networks. In: 3rd International Congress on Ultra Modern Telecommunications and Control Systems and Workshops, ICUMT (2011)
11. Ma, C., Yang, Y.: A Battery aware scheme for energy efficient coverage and routing in wireless mesh networks. In: Global Telecommunications Conference, Globecom (2007)
12. Paramanathan, A., et al.: Energy consumption model and measurement results for network coding-enabled IEEE 802.11 meshed wireless networks. In: IEEE 17th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks, CAMAD (2012)
13. Feeney, L.M., Nilsson, K.M.: Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In: Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, Infocom (2001)
14. Jamakovic, A., Braun, T., Staub, T., Anwander, M.: Authentication and Authorisation Mechanisms in support of Secure Access to WMN Resources. In: The Fouth IEEE Workshop on Hot Topics in Mesh Networking, HOTMESH (2012)

An Evolutionary Based Dynamic Energy Management Framework for IP-over-DWDM Core Networks

Xin Chen(✉) and Chris Phillips

School of Electronic Engineering & Computer Science,
Queen Mary University of London, London, UK
{Xin.Chen, Chris.Phillips}@eecs.qmul.ac.uk

Abstract. The energy consumption of the Internet accounts for approximately 1% of the world's total electricity usage, which may become the main constraint for its future growth of the Internet. In response, we propose a novel dynamic energy management framework that reduces the overall energy consumption without degrading network performance. The main concept is to combine infrastructure sleeping with virtual router migration. During off-peak hours, the virtual routers are moved onto fewer physical platforms and the unused physical platforms are placed in a sleep mode to save energy. The sleeping physical platforms are awakened again during busy periods. In our energy management framework, an important question that is considered is where to move the virtual routers to. To this end, we develop an evolutionary algorithm to solve the destination physical platform selection problem.

Keywords: Multi-objective evolutionary algorithm · Energy efficiency · IP over DWDM core networks

1 Introduction

By the end of 2016, the annual global IP traffic will be greater than one zettabyte reaching 110.3 exabytes per month [1]. This trend is driven by the growth of the customer population and ongoing development of all forms of the Internet based services, especially video streaming (IPTV, P2P, VoD etc.). The unprecedented development of the Internet brings new challenges to the Internet Service Providers and telecom companies. More capable and power-hungry network equipment is needed for supporting the increasing traffic demand. The energy consumption of the Internet accounts for about 1% of the world's total electricity [2]. It is anticipated that this will increase notably over the next couple of years. Without any new energy efficiency approaches, the energy consumption may become the main constraint on further Internet growth [3]. In response, we propose a novel dynamic energy management framework for reducing the overall network energy consumption without degrading network performance.

The Internet can be regarded as being composed of three main parts: Tier-3 access networks, Tier-2 metropolitan area networks and Tier-1 wide area networks. Our primary focus is on Tier-2 networks due to their prevalence and the transport technologies they employ [4]. More precisely, our research focuses on optical networks employing an IP over Dense Wavelength Division Multiplexing architecture.

One of the most promising energy efficiency approaches is infrastructure sleeping, which enables equipment to be switched off in the off-peak periods. The traffic demand in a core network typically has a regular diurnal pattern based on people's activities, which is high in working hours and much lighter during hours associated with sleep [5, 6]. The minimum traffic is generally about 20% to 30% of the peak traffic load. The early position paper [7] showed that infrastructure sleeping is a feasible energy saving approach by examining the impact of sleeping on several switching and routing protocols. Later, another work by Sergui et al. [8] discussed and evaluated several simple sleeping schemes by testing them with some real-world traffic workloads and topologies. In [9], the researchers proposed heuristic algorithms for switching off as many as nodes and links as possible based on different criteria, such as the number of nodes and the link utilization under various constraints. The traffic flows are rerouted via the remaining working nodes and links. If the network cannot maintain the performance after a node or a link is switched off, the node and the link is left powered on. These algorithms are off-line algorithms requiring traffic demand information. The same authors applied a heuristic algorithm based on the equipment power consumption of an actual Italian telecom core network topology and traffic demand data [10]. The results showed that the new scheme improved the network energy efficiency up to 34% during off-peak periods. However, the major disadvantage of infrastructure sleeping is that when a router is switched off, the router loses the ability to exchange routing protocol signalling messages. In other words, the logical IP-layer topology changes when a node disappears. In consequence, it triggers a series of reconvergence events that can cause network discontinuities and disruption. Therefore, we adopt a virtual router migration approach [11, 12], hiding the changes in the underlying layer whist effectively turning off physical router instances. In our previous work [13], a GA-based algorithm VRM_GA was presented for selecting the appropriate destination physical platform. In this paper, we develop a new algorithm called Virtual Router Migration – Multi Objective Evolutionary Algorithm (VRM_MOEA), for solving the destination physical platform selection problem.

The main improvements in this paper with respect to [13] are as follows. First of all, we use a multi-objective evolutionary methodology to compare two objectives separately: power consumption and virtual router migration cost, instead of combining the two objective values together into a weighted fitness function to get a specific solution. Secondly, VRM_MOEA generates a group of good solutions in a single simulation run rather than one solution as in a VRM_GA simulation run. Since the optical resource availability is not considered in the destination physical platform selection algorithm, due to computational complexity reasons, a group of good solutions provides more virtual router migration opportunities from which an appropriate one can be selected.

The rest of the paper is organized as follows. The novel dynamic energy management framework is described in Sect. 2. Section 3 provides an introduction to the destination physical platform selection problem. The new algorithm, VRM_MOEA, is described in Sect. 4. This is followed by the simulation design and modelling. Simulation results and a discussion are presented in Sect. 5. Section 6 then concludes the paper.

2 Overall Dynamic Energy Management Framework

2.1 Overview of Network Architecture

Our research network scenario (topology, traffic, routing) is an IP over Dense Wavelength Division Multiplexing (DWDM) core network. IP over DWDM is a promising architecture for the IP-based next generation networks. It is composed of two principal layers: an IP-layer and an optical layer. This can be abstracted as a structure whereby a node is composed by a set of network equipment, e.g. a router and an optical transport switch, that are then inter-connected by an optical line system, which includes optical fibres and amplifiers. In our architecture, we assume that nodes are composed of an IP core router and a Reconfigurable Optical Add-Drop Multiplexer (ROADM) that are interconnected by a transmission line system.

There are effectively two networks in our research: a substrate network and a virtualization network. In the substrate network, each node is a physical platform (PP) providing the hardware support for one or more virtual router instances. Above this, a virtualization network exists, which is composed of a set of virtual routers (VRs) which manage, configure and monitor the routing functionalities. A PP can support several VRs and the substrate network can be shared by several virtualization networks. In our research, we only assign one virtualization network onto the substrate network.

2.2 The Overall Procedure

In this section, we present the overall procedure of the novel dynamic energy management framework. The principle steps are as follows:

1. **Collect and analyze the network conditions**. Typically we monitor the state of the network at 5-minute intervals. We then use a reactive mechanism to trigger the VR migration, as appropriate. The information collected in this stage is used for determining whether the system state matches the conditions necessary for migration. The information includes the network traffic load, the PP utilization and optical resource availability. It is necessary to determine which VRs are viable candidates to be migrated so that PP(s) can be placed in a sleep state or to determine when PP(s) need to be re-awoken to accommodate VR(s).
   A PP has three states: *Quiet*, *Normal* and *Busy*. Quiet and Busy thresholds are used to distinguish these three states. We consider a PP has a light workload if the PP utilization is lower than the Quiet threshold. In this condition, we may consider consolidating VRs onto fewer PPs; PPs not accommodating any VRs can then be placed in the sleep state. If PP utilization is higher than the busy threshold,

the VRs are moved away from the busy PPs. If the utilization is between Quiet and Busy thresholds, no change is required. If the system matches the migration conditions, go to the step 2.
2. **Select the appropriate destination PPs** for the VRs based on the new VRM_MOEA algorithm that aims to maximize the consolidation of VRs onto as few PPs as possible, given various constraints. VRM_MOEA is described in Sect. 4. The optical resource availability is tested in this step.
3. **Establish new optical connection(s)**. After obtaining a good solution for VR destination locations, some new optical connections are established for transmitting the traffic to the remotely relocated VRs when VR migration finishes.
4. **Virtual router migration**. In this step, the VR(s) are moved to their appropriate destination(s) based on the identified solution.
5. Where appropriate, **switch off (on) the corresponding PPs** and remove the unneeded optical connection(s).
6. Go back to step 1 to **recheck the network conditions**.

3 Destination Physical Platform Selection

3.1 Destination Physical Platform Selection Problem

An important question in our dynamic energy management framework is where to move the VRs to, given various constraints. We call this the destination physical platform selection problem. It is to determine which VRs are viable candidates to be migrated so that the PP can be placed in a sleep state or to determine when a PP needs to be re-awoken ready to accommodate a VR instance.

A reactive mechanism is used for collecting the information of the network status. The information collected is used for judging whether the system meets the conditions suitable for migration. When the system discovers that the condition exists, we need an algorithm to find a good solution in an acceptable period of time. The algorithm has two objectives. The first one is to reduce energy consumption during the off-peak hours. We want to maximize the consolidation of VRs onto as few PPs as possible for a given traffic demand without degrading the network performance. Where appropriate, we therefore switch off surplus PPs to save energy. Secondly, we also want to minimize the virtual router migration cost. For simplicity, the hop-distance is used to determine the VR migration cost. These two objectives are considered further in Sect. 3.2.

3.2 Two Objective Functions: Power Consumption and Migration Cost

The two performance metrics used for assessing candidate solutions are the power consumption of the network equipment and the VR migration cost. Our goal is to minimize the network power consumption whilst minimizing the migration cost, given various constraints.

1. Power Consumption

Power is consumed by both node and line equipment. In practice, the power consumption of optical line equipment can be omitted [14]. Therefore, we only consider the power consumption of node equipment. Node equipment includes the PPs hosting the VRs and the ROADMs.

The power consumption of a PP comprises two sections: a base system and a number of line cards. The power consumption of a working PP P_{pp} can be represented as:

$$P_{pp} = P_{base} + N \cdot P_{lc} \tag{1}$$

where P_{base} denotes the base system power consumption. N is the number of active line cards on that PP and is dynamic depending upon the VR requirements being hosted on the PP. P_{lc} is the power consumption of a line card.

For an ROADM, it is always working even if the PP it connected to is sleeping. Thus in our model, the ROADM is assumed to be always powered on. The overall power consumption P_{total} is:

$$P_{total} = \alpha \cdot P_{pp} + \theta \cdot \ (\beta - \alpha) \cdot P_{base} + \beta \cdot P_{roadm} \tag{2}$$

where α is the number of active PPs and β is the number of ROADMs in the network. θ stands for a percentage of the base system power consumption a PP consumes when it is sleeping. We assume a sleeping PP consumes 5% of the base system power consumption and one PP is always associated with the same ROADM. Combining (1) with (2), P_{total} can be expressed as:

$$P_{total} = \alpha \cdot P_{base} + \sum_{i=1}^{\alpha} N_i \cdot P_{lc} + \theta \cdot \ (\beta - \alpha) \ \cdot P_{base} + \beta \cdot P_{roadm} \tag{3}$$

where β is the number of ROADMs in the network, N_i is the number of active line cards in the i-th PP.

2. Virtual Router Migration Cost

We assume there is a cost for VR migration. In this investigation, the cost of migration is only considered in terms of the additional optical resources consumed when having to forward traffic to a remote PP. We take into account the finite nature of these resources, given the wavelength continuity constraint and the limited number of optical channels per link. However, we do not currently take into account the time needed to transfer the state information of the VR instances. Instead we assume that this information can be transferred in a negligibly small time. In practise research suggests that this may typically take a few seconds [12]. For simplicity, we measure the migration cost in terms of hop-distance. In our system the VR migration cost comprises two components. They are described in the following part.

The first migration cost component considers the propagation delay from an original VR location on the default PP to its destination PP. We assume every VR has a default PP and a default ROADM. The traffic comes from an access network is always processed by a particular VR. If a VR moves far away from its default PP, some additional optical connections are needed for re-directing the data traffic to the remotely located VR. Therefore, the longer the distance between a default PP and a destination PP, the larger the cost is.

We obtain the distance between two PPs by the function *d* (x_1, x_2), where x_1, x_2 denote the two PPs concerned. The first migration cost component *Cost_a* for a VR is

$$Cost\_a = d(x_0, x_d) \tag{4}$$

where x_0 is the default PP for a VR and x_d is the destination PP.

The second cost component takes into account the current VR location. If the source and destination PPs are far away from each other, it will take a long time for transmitting the control plane information to the destination PP. The second cost component is based on the physical distance between the source and destination PP. The second cost component for a VR is *Cost_b*:

$$Cost\_b = d(x_{current}, x_d) \tag{5}$$

where $x_{current}$ is the source PP. The overall cost of a moved VR is:

$$Cost = \varphi \cdot Cost\_a + (1 - \varphi) \cdot Cost\_b \tag{6}$$

where φ is a weight of two migration cost components.

3.3 Constraints

Several constraints that need to be considered as follows:

1. The source and destination PP should be compatible with each other since different manufacturers use different router operating systems. If two PPs are not compatible, the VR may not be able to run on the destination PP. In our research, we assume that all the PPs are homogeneous.
2. The destination PP should be able to accommodate the new VR(s) without detrimentally impacting on the performance of any VRs it is currently hosting.
3. The optical resource supports the new network configuration. We consider the wavelength continuity constraint in the framework so O-E-O conversion is not permitted in pass-through connections. However, this requires there to be sufficient optical channels to transit the traffic flows to the remote VR(s); if not, the migration cannot take place.

4 New Evolutionary Algorithm: VRM_MOEA

In this section, we discuss why we use a multi-objective evolutionary approach to solve the destination Physical Platform selection problem. The following sections describe how VRM_MOEA works.

4.1 Why a Multi-Objective Evolutionary Algorithm

For the destination PP selection problem, we need an algorithm that meets the following requirements:

1. For the selection problem, we use two metrics to measure the possible new destinations. These two objectives are neither totally consistence nor in competition. A good solution depends on the different scenarios. For example, a solution that gives very small power consumption may also have a large VR migration cost during the off-peak hours. In the peak periods, the good solution may have greater power consumption and but a larger migration cost. The best we can do is to find an approach that can handle the different scenarios to obtain reasonable solutions.
2. We use a reactive mechanism to trigger the VR migration. The algorithm should be able to provide a reasonable good solution in a relatively short period, such as 5 min or less.
3. We do not consider the optical resource availability in the algorithm. The optical resource availability limits the possibility of VR migration. Therefore, the algorithm should generate a group of viable solutions giving more opportunities for VR migration. Optical resource availability is considered when selecting a solution from these candidates.

Considering the features above, we choose a methodology based on multi-objective evolutionary algorithm. The evolutionary algorithm represents a class of stochastic optimization methods that use biologically inspired mechanisms like mutation, crossover, natural selection, and survival of the fittest in order to refine a set of solution candidates iteratively. Multi-Objective Evolutionary Algorithms (MOEA) use evolutionary methodologies to solve the problems involving multiple conflicting objectives and an intractably large and highly complex search space. The goal of MOEAs is to find a group of trade-off solutions called a Pareto-optimal solution set or Efficient Frontier. The user can then choose a preferred solution from the set using higher-level considerations.

All MOEAs work on a set of candidate solutions called a population. By employing two basic principles: selection and variation, the population is modified generation after generation. The selection mechanism mimics the fierce competition for surviving in the natural world. The fittest individuals have a greater chance of survival and produce offspring. Variation includes mutation and crossover genetic operators that imitate the process of producing offspring, sharing the genetic information between parent's chromosomes and changing the gene randomly to some extent.

We now proceed to develop a new algorithm called VRM_MOEA for finding a group of reasonable solutions. The operations of VRM_MOEA are also described in the following sections.

4.2 Individual Chromosome Representation

We use a decimal representation because it is straightforward and more natural. An individual stands for one solution indicating a possible mapping of VRs onto PPs. The number of VRs determines an individual's length. A gene index indicates the name of a VR and the allele (numeric value of each gene) represents the name of the PP that holds the corresponding VR. An example representation of an individual "1122" is shown below. It is clearly that there are 4 VRs and 4 PPs in the network. VR1 and VR2 are running on the PP1 because gene 1 and gene 2 are 1. By the same rule, we can know VR3 and VR4 are running on PP2 (Fig. 1).

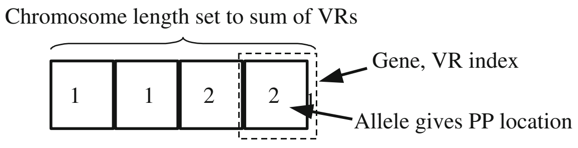


Fig. 1. An Example Individual Chromosome

4.3 Fitness Function

A fitness function is used for measuring the quality of the individuals. In each generation, every individual is assigned a fitness value, which stands for its quality in the current population. According to the fitness values, some "bad" individuals are eliminated from the population and "good" ones survive.

For fitness assignment, we employ the SPEA2 method described in [15]. SPEA2 is a multi-objective evolutionary algorithm that has been shown to have very good performance for solving two-objective optimization problems. An individual fitness is evaluated by comparing all objective values separately with the rest of individuals in the population. The advantage of SPEA2 is that introduces an external population called the "elite" archive to retain the nondominated solutions. It is an important feature as it guarantees that the final solutions are nondominated with respect to all other solutions in the evolutionary process rather than the current population.

4.4 Selection Mechanism

The selection mechanism chooses some individuals from the current population to be parent individuals in the mating pool to reproduce the offspring. Popular selection mechanisms include fitness proportionate selection and tournament selection. We use tournament selection in VRM_MOEA which chooses the winners in the tournaments whose participants are chosen randomly from the population.

4.5 Crossover Operation

The crossover mechanism is used to share genetic information from two or more parents to generate the offspring. The rate of crossover is controlled by a crossover probability. In the binary representation, there are many kinds of crossover such as one-point crossover, two-point crossover and uniform crossover. For the real coded (such as decimal and floating point representation) algorithms, the performance of traditional crossover methods is poor. Therefore, many efficient real-coded crossover operators, such as arithmetical crossover, geometrical crossover and BLX-α crossover [16, 17] have been proposed. In VRM_MOEA, we use BLX-α crossover.

4.6 Mutation Operation

The mutation mechanism is used for maintaining the genetic diversity from one generation to another. For each gene in a chromosome, a uniform random number is generated between the interval [0, 1]. If the random variable is smaller than the user-defined mutation probability, the gene can be modified. Mutation operators may alter one or more gene values in a chromosome.

5 Simulation Modelling

There are various ways the new framework could be simulated including the use of existing commercial software or building a new specific simulator. Since our framework is novel, no suitable protocol or architectural models exist in the commercial software; these would have to be created. However, we also needed a means of simulating packet flows over high-speed optical links over many hours. This is not feasible with a traditional packet-level simulator. Instead we constructed a new hybrid fluid flow/packet simulator that could achieve this and possess all the features we needed. This simulation tool was created using C++ and enabled us to evaluate the performance of the new dynamic energy management framework over many hours of simulated time.

The data traffic is modelled as time-varying flows. Although traffic is stochastic in nature giving rise to significant variation about the mean flow rate, in our case, we are dealing with aggregated flows over a relatively long time-frame. Under these circumstances it is common to represent the traffic using a fluid approximation [18, 19]. We assume that for each pair of peering nodes in the network, they have a similar traffic pattern. For simplicity, we used a sinusoidal function [9] to model the changes of traffic flows. The optical channel resources are also modelled in the simulator. We assume that optical resources are finite (i.e. wavelengths per channel) and the wavelength continuity constraint is applied.

The power consumption data are taken from the existing commercial products. Table 1 shows the power consumption of the equipment [20–22].

Table 1. Power Consumption of the Network Architecture

Parameter	Power Consumption (W)
Router Chassis (P_{base})	5800
Line Card (P_{lc})	550
ROADM (P_{roadm})	350

6 Simulation Results and Discussions

In this section, we evaluate the energy saving percentage of the new dynamic energy management framework. Since there is no previous work on the destination physical platform selection problem, we develop a quick and easy algorithm called Quick VRM to compare the performance of VRM_MOEA. The main idea of Quick VRM is to choose the best solution in a randomly generated population of candidate solutions without any evolutionary process, e.g. selection, crossover and mutation. The new dynamic algorithm is expressed as VRM_MOEA *(a,b)*. *a* and *b* stand for the weighted parameters of two cost terms *Cost_a* and *Cost_b*. The sum of *a* and *b* is 1. We use five different settings of *a* and *b* to measure the impact of the two cost components.

Table 2 shows the daily energy consumption and energy saving in a 6-node-8-link (6N8L) and 11N14L network. Each PP capacity is 1.2 Tbps which is similar to the Cisco CRS-1 [21]. We assume that each fibre has 40 optical channels with 40Gbps capacity per channel. The energy consumption is an average value over 5 simulations with random seeds. In VRM_MOEA, the number of population is 40 in each generation and crossover rate is 0.9 and mutation rate is 0.1. The stop generation is 2000 in the 6N8L network and 10,000 in the 11N14L network. The CPU execution time is around 2 s in the 6N8L network and 8 s in the 11N14L network which is less than the 5-minute monitoring interval.

We can see the energy saving is similar among the different VRM_MOEA schemes in the two networks. VRM_MOEA schemes save more energy than Quick VRM. The energy saving between VRM_MOEA and Quick VRM increases with the network size. The Quick VRM has good performance since the two networks are not very large and the result in Table 2 only reflect the energy issue. In a complex network, the Quick VRM may increase risk of the connection blocking rate. Therefore, we observe the energy saving for the different schemes in Fig. 2 to determine when VRM_MOEA schemes obtain more energy saving than that of Quick VRM. We can see clearly that in the peak hours, e.g. hour 1 to 12, there is little difference among all schemes. In the off-peak hours, e.g. 12 to 24, the energy saving of VRM_MOEA achieves 50% while Quick VRM's saving is around 35%. It means VRM_MOEA yields better solutions than Quick VRM. The evolutionary process is of benefit in the destination physical platform selection problem.

Table 2. Energy Consumption of Different Schemes in a 6N8L and an 11N6L Network

Network	6N8L		11N14L	
Scheme Name	Energy Consumption (A day)	Energy Saving	Energy Consumption (A day)	Energy Saving
No VRM	4057200.00	0.00%	7698240.00	0.00%
Quick VRM	3223327.20	20.55%	6008048.00	21.96%
VRM_MOEA(0 ,1)	3134507.20	22.74%	5457832.20	29.10%
VRM_MOEA(0.2,0.8)	3125365.00	22.97%	5451220.20	29.97%
VRM_MOEA(0.5,0.5)	3140139.80	22.60%	5471056.80	28.93%
VRM_MOEA(0.8,0.2)	3160408.40	22.10%	5490892.40	28.67%
VRM_MOEA(1 ,0)	3144768.20	22.49%	5517340.00	28.33%

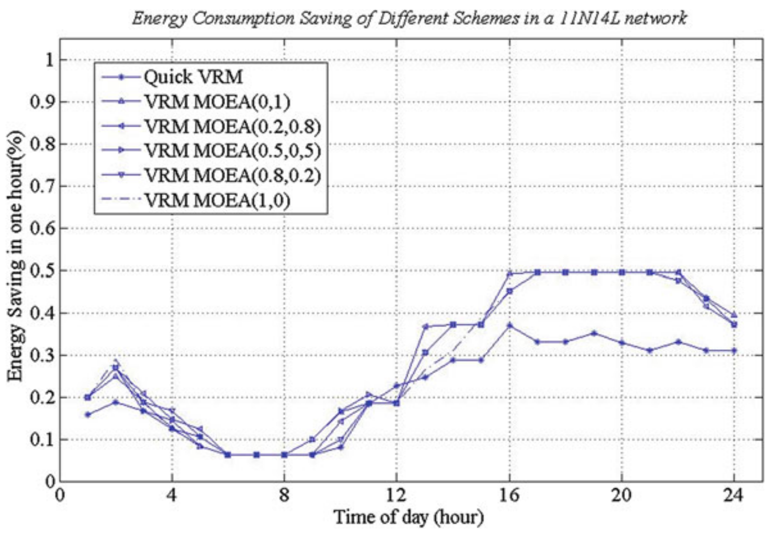


Fig. 2. The Energy Saving of Different Schemes in an 11N14L Network

In Fig. 3, we explore the number of the occupied optical channels in the baseline scheme (without VRM), Quick VRM and different VRM_MOEA schemes. The occupied number is recorded every 5 min. It is clear that the occupied number fluctuates with the traffic load in the baseline scheme. In other schemes, the discontinuities on the lines correspond to the VR migrations. The small changes of occupied number indicate the traffic load variation. In other schemes, when VRs are moved to remote

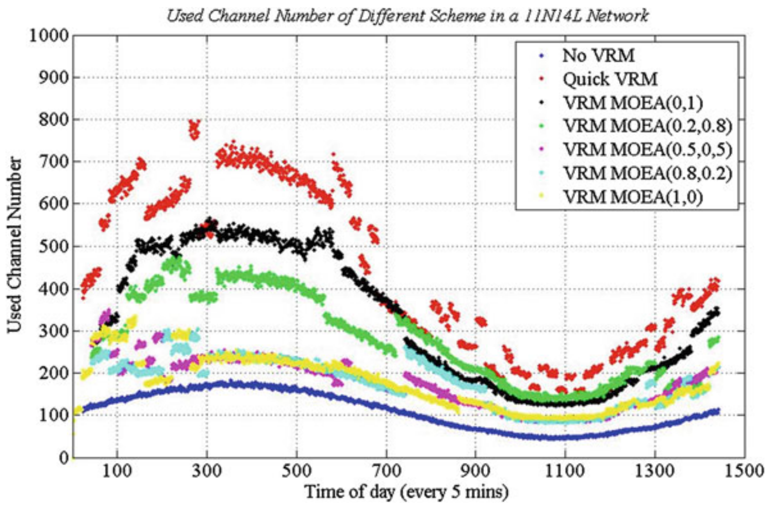


Fig. 3. The Occupied Number of Optical Channel of Different Schemes in an 11N14L Network

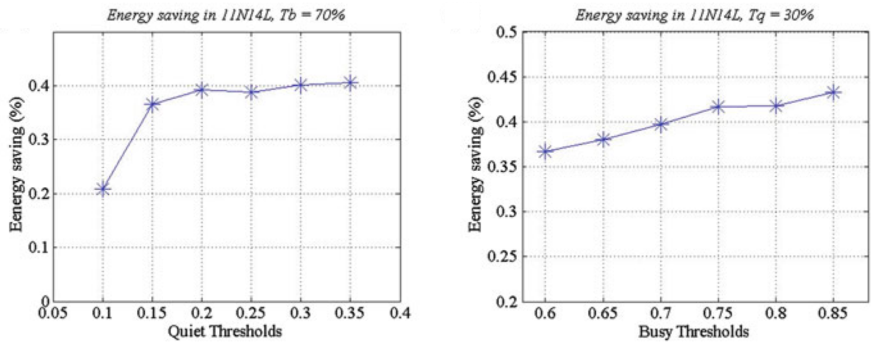


Fig. 4. Threshold Impact (a) Quiet Threshold Impact (b) Busy Threshold Impact

PPs instead of their default place, more optical channels are used for transmitting the packets to be processed by VRs than that of the baseline case. Since Quick VRM does not consider any difference between the default PP location, current PP location and destination PP location, it has the highest optical channel usage which may increase the risk of blocking. In the VRM_MOEA algorithm, the *Cost_a* term represents the distance from default location to the destination PP. When the weighted parameter of *Cost_a* increases, the occupied channel number gets closer to the baseline scheme. This is because the algorithm is searching for the solutions that are close to the VRs' default location which reduce the number of additional optical channel from default PP to destination PP.

We investigate the effect of the Quiet and Busy thresholds in Fig. 4. We use five 11N14L networks with the randomly generated topologies. The traffic pattern for all VR pairs is similar. Figure 4a shows the variation of the energy saving percentage versus the Quiet threshold with a particular Busy threshold. It indicates that the energy saving percentage goes up with the increasing Quiet threshold. This is due to a higher Quiet threshold allowing the PP enters the quiet state for longer periods than with a lower Quiet threshold. Therefore, a longer consolidation time leads to greater energy

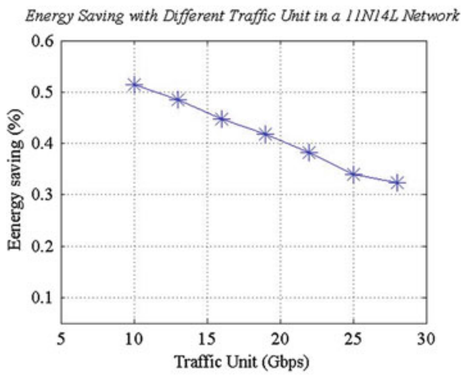


Fig. 5. Energy Saving with Different Average Traffic Amount in the 11N14L Networks

saving. Similarly, for the Busy threshold impact, the higher the Busy threshold the bigger the energy saving percentage (Fig. 4b). This is because a lower Busy threshold lets PPs enter the busy state sooner, which shortens the consolidation periods.

From the results in Fig. 4, it seems that high Quiet and Busy thresholds can bring larger energy saving. However, we currently use a reactive mechanism with interval observations. A high Busy threshold may bring with it the risk of traffic loss. For example, consider a PP accommodating several VRs in a scenario with a high busy threshold. Assume that during the previous observation time, the PP utilization is near to the busy threshold. If the traffic increases quickly and exceeds the maximum PP capacity before the next observation time, some traffic between two observation times may be lost as we have reached the forwarding capacity of the PP switch fabric.

We also observe the impact of the traffic load on the framework performance in Fig. 5. A similar daily traffic pattern is used for each VR pair. The larger the average traffic load, the busier the network. We set the off-peak period traffic to be 20% of the peak load. We perform the simulations on five 11N14L networks with the different topologies. For each measured average traffic value, we repeat the simulations 5 times with different random seeds. For the sake of simplicity, only the average values are shown. The Quiet threshold is 0.3 and the Busy threshold is 0.8. It is clear to see that with a higher traffic load, the energy saving percentage decreases. It implies that in a busier network, it is more difficult to obtain energy saving using our framework.

7 Conclusion

In this paper, we propose an evolutionary algorithm based dynamic energy management framework for IP over DWDM core networks. We combine infrastructure sleeping and virtual router migration approaches to reduce the overall network energy consumption during the off-peak periods. VR migration is used for hiding changes in the IP-layer topology to avoid the discontinuities and service disruption when PPs enter or leave their sleep state.

The two significant questions are when to trigger the VR migration and where to move to the VRs to, given various constraints. Currently, we use a reactive mechanism to trigger the VR migration by monitoring the network state. If the network state satisfies the conditions for migration, we perform operations such as consolidating VRs onto fewer PPs or moving VR(s) from busy PP(s). We also propose a new algorithm called VRM_MOEA to solve the destination physical platform selection problem. In VRM_MOEA, a new individual representation and dual objective functions are developed.

The paper also provides details of the dynamic energy management framework and the simulation models that we constructed. Simulation results show that the performance of network energy saving depends on many factors, such as network topology, quiet and busy thresholds, and traffic load; however, savings of around 30% are possible with typical medium-sized network topologies.

Acknowledgment. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

References

1. Cisco Press Release: Cisco Visual Networking Index: Forecast and Methodology, 2011–2016. http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/ white_paper_c11-481360.pdf
2. Tucker, R.S.: Energy consumption in telecommunications. In: 2012 IEEE Optical Interconnects Conference, pp. 1–2, 20–23 May (2012)
3. Caria, M., Chamania, M., Jukan, A.: To switch on or off: a simple case study on energy efficiency in IP-over-WDM networks. In: IEEE 12th International Conference on High Performance Switching and Routing (HPSR), 2011, pp. 70–76, 4–6 July 2011
4. Baliga, J., Hinton, K., Tucker, R.S.: Energy consumption of the Internet. In: COIN-ACOFT 07, pp. 1–3 (2007)
5. Uhlig, S., Quoitin, B., Lepropre, J., Balon, S.: Providing public intradomain traffic matrices to the research community. SIGCOMM Comput. Commun. Rev. **36**(1), 83–86 (2006)
6. Abilene Network Traffic Statistics. [Online]. http://www.abilene.iu.edu/
7. Gupta, M., Singh, S.: Greening of the Internet. In: Proceedings of ACM SIGCOMM'03, Karlsruhe, Germany, August 2003
8. Nedevschi, S., Popa, L., Iannaccone, G., Ratnasamy, S., Wetherall, D.: Reducing network energy consumption via sleeping and rate-adaptation. In: Proceedings of NSDI'08, 5th USENIX Symposium on Networked Systems Design and Implementation, pp. 323–336 (2008)
9. Chiaraviglio, L., Mellia, M., Neri, F.: Reducing power consumption in backbone networks. In: ICC'09, pp. 1–6 (2009)
10. Chiaraviglio, L., Mellia, M., Neri, F.: Energy-aware backbone networks: a case study. In: ICC Workshop 09, pp. 1–5 (2009)
11. Agrawal, M., Bailey, S.R., Greenberg, A., Pastor, J., Sebos, P., Seshan, S., Van Der Merwe, K., Yates, J.: RouterFarm: towards a dynamic, manageable network edge. In: Proceedings SIGCOMM Workshop, INM'06, pp. 5–10 (2006)
12. Wang, Y., Keller, E., Biskeborn, B., Van Der Merwe, J., Rexford, J.: Virtual routers on the move: live router migration as a network-management primitive. In: Proceedings SIGCOMM'08, pp. 231–242 (2008)
13. Chen, X., Phillips, C.: Virtual router migration and infrastructure sleeping for energy management of IP over WDM networks. In: 2012 International Conference on Telecommunications and Multimedia (TEMU), pp. 31–36, 30 July 2012–1 August 2012
14. Baliga, J., Ayre, R., Hinton, K., Tucker, R.: Photonic switching and the energy bottleneck. In: Photonics in Switching, pp. 125–126 (2007)
15. Zitzler, E., Laumanns, M., Thiele, L.: SPEA2: Improving the strength pareto evolutionary algorithm. In: Giannakoglou, K. et al. (eds) EUROGEN 2001. Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems, pp. 95–100, Athens, Greece (2002)
16. Herrera, F., Lozano, M., Verdegay, J.L.: Tacking real-coded genetic algorithms: operators and tools for behavioural analysis. Artif. Intell. **12**(4), 265–319 (1998)
17. Herrera, F., Lozano, M., Sánchez, A.M.: A taxonomy for the crossover operator for real-coded genetic algorithms: an experimental study. Int. J. Intell. Syst. **18**, 309–338 (2003)
18. Gu, Y., Liu, Y., Towsley, D.: On integrating fluid models with packet simulation. INFOCOM 2004. In: Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, pp. 2856–2866, 7–11 March 2004

19. Kiddle, C., Simmonds, R., Williamson, C., Unger, B.: Hybrid packet/fluid flow network simulation. In: Proceedings of the Seventeenth Workshop on Parallel and Distributed Simulation, 2003, pp. 143–152, 10–13 June 2003
20. Wang, L., Lu, R., Li, Q., Zheng, X., Zhang, H.: Energy efficient design for multi-shelf IP over WDM networks. In: INFOCOM'11, Workshop on Green and Communications and Networking (2011)
21. Cisco data sheet: Cisco CRS-1 16-Slot Single-Shelf System. http://www.cisco.com/en/US/prod/collateral/routers/ps5763/ps5862/product_data_sheet09186a008022d5f3.pdf
22. Ciena data sheet: CN 4200 ROADM. http://www.ciena.com

Autonomic Computing to Manage Green Core Networks with Quality of Service

Remi Sharrock[1(✉)], Thierry Monteil[2,4], Patricia Stolf[3,4], and Olivier Brun[2]

[1] Institut Mines-Telecom, Telecom ParisTech, CNRS LTCI UMR 5141, Paris, France
remi.sharrock@telecom-paristech.fr

[2] CNRS, LAAS, 7 avenue du Colonel Roche, 31077 Toulouse Cedex 4, France

[3] IRIT, 118 Route de Narbonne, 31062 Toulouse Cedex 9, France

[4] UPS, INSA, INP, ISAE; UT1, UTM, LAAS, Université de Toulouse, 31077 Toulouse Cedex 4, France

Abstract. In a context where data and computing services are moving to external specialized datacenters the manual management of these systems is becoming an issue. Human administrators have to deal with hardware resources optimization while meeting the users' needs. In our approach we propose to reconfigure both a set of applications deployed in a datacenter (by adapting their behaviors using autonomic computing) and the wired network (by switching on and off its equipments like routers, modules). We take into account both the energetic costs with the network equipments and the quality of service provided to the end user by the deployed applications. The main contribution of the proposed model is to consider a compromise between the total power consumption of the network equipments and the application quality of service. We validated our approach by simulating deployed applications on the Grid'Mip infrastructure similar to a small core network made up of Cisco routers (part of Grid'5000 project).

1 Introduction

Data and computing services outsourcing (Web sites, Databases, Distributed applications) towards specialized datacenters is increasingly selected as a way to delegate complex infrastructure management to experts. Some of these datacenters tend to offer cloud computing services which allow access to pre-configured platforms or software environments, sometimes in a matter of seconds. From the datacenter customer point of view, these services allow to adapt the application architecture dynamically, for example according to the fluctuating incoming request flows. From the datacenter's administrator point of view, the mix of customers' needs and deployed application's needs may require an optimization of the hardware resources usage, for example to reduce the electrical power consumption of the datacenter. Thus, there is a compromise to be found between, on one hand, the quantity of resources (power) that are needed for a given service and on the other hand the quality (performance) of this service.

DOI: 10.1007/978-3-642-40517-4_20,

In our approach, we tend to introduce the general problem of self-optimization applied at both application (performance optimization) and network infrastructure levels (power optimization). We introduce a criterion which is a function of the global electric power consumed by the network and of a parameter representing the loss of service quality for the applications. The problem of optimization consists in minimizing this loss and/or the power consumption depending on the administrator's choice. This minimization takes into account a number of constraints such as the dynamic topology (links shutdowns, network equipments or nodes shutdowns), the network links capacities and the routing policies.
The rest of the paper is organized as follows. Related work is first discussed in Sect. 1.1. Then the environment considered by our approach is presented in Sect. 1.2. Section 2 presents the model proposed including the network, the application, the routing protocol and the criterion used. Results of our experimental evaluation are presented in Sect. 3 , then finally, we conclude in Sect. 4.

1.1 Related Work

Quality of Service characteristics The QoS characteristics are measurable values and a set of constraints on these values. They are mainly classified in the following groups [1]:

- **Temporal**: transit time, response time, time of establishment, jit
- **Capacity**: entrance/exit flow rate, user data rate, data control rate
- **Integrity**: connection breakdown probability, loss probability
- **Security**: protection level, authentication

Difference between QoS characteristic, profile and needs It is important to first differentiate the **QoS profile** of an application (its multiple behaviors) and the **QoS characteristics**. Indeed, the first one has an influence on the other. The QoS profile is a set of internal values for the application which influences the QoS characteristics. For example, a video application may have two possible types of compression for the images [2], one using the BZIP2 algorithm and the other Lempel-Ziv. The internal parameter is the compression algorithm which influences the transmission time because both algorithms have different compression times. We can thus say that the QoS profile of the application is described by the compression algorithm (two behaviors possible) and the influenced QoS characteristics is the transmission time. In this case, by using the BZIP2 algorithm, the application has a transmission time of 20 ms, whereas with the algorithm Lempel-Ziv the transmission time is 40 ms. Finally, **QoS needs** are a set of predefined QoS characteristics needed by the application. Depending on its different QoS profiles, the application may have different QoS needs.

A self-adaptive application would choose automatically its QoS profile so as to satisfy the QoS needs, which is a complex optimization problem by itself. In our approach, we introduce a mathematical model which takes into account those QoS needs. One part of the optimization consists in selecting which QoS needs will satisfy at best the global minimization problem.

Applications' QoS management capacity We consider two cases for the applications' QoS management capacity: either the application has multiple QoS profiles (multiple behaviors) or it has absolutely no notion of QoS (no QoS profile at all and only one behavior).

Application without notion of QoS: for an application without notion of QoS, it cannot change its functioning neither at the beginning, nor during the execution.

Applications with multiple QoS profiles: the application has the capacity to modify its behavior and is able to send/receive different traffics (for example, compression of different data). In this case, either the application is self-adaptive and has internal mechanisms to choose its QoS profiles or the application is external-adaptive and rely on external mechanisms to choose its QoS profile.

For our approach, we consider external-adaptive applications with multiple QoS profiles. Indeed, the change of QoS profile (the reconfiguration) is made by an autonomic tool according to the results of the optimization. This allows to make reconfiguration decisions at the global level, by taking into account all the applications and the datacenter network infrastructure.

1.2 Environment Considered by the Approach

The DiffServ domain Our approach considers one DiffServ domain[3]. The DiffServ model is implemented in a DiffServ domain (DS Domain) which corresponds to a zone having a common QoS policy, usually true for datacenter network. Indeed, a datacenter is for the most part managed by a single administrative entity which integrates the network into a DS domain belonging to the entity. Our approach proposes to use a single optimization tool for the DS Domain.

The routing of the network Usually, the machines of a datacenter are grouped in clusters. A cluster can be connected physically to several routers. This means that all the machines of the cluster are connected with several routers and thus have several network cards. Our approach considers only the case of the dynamic routing tables and does not take into account the case of Channel Bonding[4] (aggregation of several network interfaces in a logical interface).

Also, for the border routers and core routers, the routing is either static, or it depends on the routing protocol used inside the DS domain, meaning that it depends on the IGP (Interior Gateway protocol). The roles of an IGP are:

- to establish the optimal routes between a point of the network and all the destinations available of a bounded domain;
- to avoid buckles;
- in case of modification of the topology (disconnection of a physical link, a router's shutdown), to guarantee the convergence of the network, that is the restoring of it's optimal connectivity without buckle as soon as possible.

We distinguish usually:

– link state protocols which establish neighborhood tables and use the Dijkstra algorithm to calculate the best routes. Two examples of such protocols are *IS-IS* (Intermediate system to intermediate system) [5], *OSPF* (Open Shortest Path First) [6];
– distance vector protocols: *RIP* (Routing Information Protocol), *IGRP* (Interior Gateway Routing Protocol) [7];
– The hybrid protocols, which have characteristics of both first ones: *EIGRP* (Enhanced Interior Gateway Routing Protocol) [8].

The inclusion of QoS and network energy consumption can be done at the protocol level. [9] proposes the use of energy Efficient Ethernet standard (IEEE 803.3az) by adding a prioritization of streams impacted by the saving energy mechanisms (sleeping mode, coalescing mechanism) to ensure the desired level of QoS. In [10], is proposed a change in the OSPF that allows (depending on the communication links usage) to reconfigure the routing tables and to turn off routers. In [11], the authors present some detailed router consumptions and a generic router consumption model. Studies specifically on datacenter and core networks have been made in [12]. They are interested in the location of datacenters from the data access point of view under network energy consumption. The authors also define classes of popularity data leading them to address the problem of data replication. There is a linear programming formulation to find a solution.

Our approach considers the case of various routing protocols. We give a first example of heuristics with a dynamic routing table generated by the OSPF routing protocol. The choice of OSPF as routing protocol has for consequence to create routing tables following a metric defined on the links. For example, the cost of routes can be calculated according to the capacities of all the links of the route. A load balancing is made on routes having the same cost and routes having a higher cost are not used. It is then possible to switch off some router links or some routers.

2 Model

2.1 Network Model

Variables and functions for the network representation The topology of the network is expressed with an oriented graph $G = \{\mathbf{N}, \mathbf{E}\}$ where $\mathbf{N}$ is the set of nodes of one network domain and $\mathbf{E}$ is the set of edges. A couple (i, j) represents the edge between the node i and the node j. A node can be a border router, a core router or a host. A router has different modules plugged into his frame. Each module contains several network ports that can be used to create links between the nodes. We consider that the hosts have only one module but possibly multiple network ports.

– $\mid \mathbf{N} \mid = n_N$ and $\mid \mathbf{E} \mid = n_E$
– $c^e_{i,j}$ is the capacity of the edge (i, j)

- $p^e_{i,j}$ is the electric power consumption of the port in the node i used to create the link between the node i and the node j
- $z^e_{i,j}$ defines the state of the port in the node i used for the link (i,j), 1 means the port is switched on and 0 is switched off
- $p^m_{i,l}$ is the electric power consumption of the module l of the node i
- $z^m_{i,l}$ defines the state of the module l of the node i (same convention as for $z^e_{i,j}$)
- p^c_i is the electric power consumption of the node i when all modules and ports are switched off (also called the frame consumption)
- z^c_i defines the state of the node i (same convention as for $z^e_{i,j}$)
- $\mathbf{M_i}$ is the set of the modules in the node i :
  $\mathbf{M_i} = \{m_1, \dots m_l \dots, m_{n_{M_i}}\}, \mid \mathbf{M_i} \mid = n_{M_i}$
- $\mathbf{E_{i,l}}$ is the set of ports of the module l of the node i
- $\mathbf{E_i}$ is the set of all the ports of the node i, $\forall i \in \mathbf{N}$:
  $\mathbf{E_i} = \bigcup_{l \in [1, n_{M_i}]} \mathbf{E_{i,l}}$

Relations and constraints for the network The total electric power consumption of the network P_{total} is:

$$P_{total} = \sum_{i \in \mathbf{N}} \{p^c_i . z^c_i + \sum_{l \in [1, n_{M_i}]} [p^m_{i,l} . z^m_{i,l} + \sum_{j \in \mathbf{E_{i,l}}} p^e_{i,j} . z^e_{i,j}]\} \tag{1}$$

$$= \sum_{i \in \mathbf{N}} p^c_i . z^c_i + \sum_{i \in \mathbf{N}, l \in [1, n_{M_i}]} p^m_{i,l} . z^m_{i,l} + \sum_{i \in \mathbf{N}, j \in \mathbf{E_i}} p^e_{i,j} . z^e_{i,j} \tag{2}$$

If all ports of a module are switched off the module can also be switched off:

$$\forall i \in \mathbf{N}, \forall l \in [1, n_{M_i}] : \sum_{(i,j) \in E_{i,l}} z^e_{i,j} = 0 \Rightarrow z^m_{i,l} = 0 \tag{3}$$

Because it is easier to solve a linear problem the relation (3) could be expressed as two linear constraints:

$$\forall i \in \mathbf{N}, \forall l \in [1, n_{M_i}] : z^m_{i,l} - \sum_{j \in \mathbf{E_{i,l}}} z^e_{i,j} \leq 0 \tag{4}$$

$$\forall i \in \mathbf{N}, \forall l \in [1, n_{M_i}], j \in \mathbf{E_{i,l}} : z^e_{i,j} - z^m_{i,l} \leq 0 \tag{5}$$

The same linear constraints can also be written for the nodes. If all modules of a node are switched off the node can also be switched off:

$$\forall i \in \mathbf{N} : z^c_i - \sum_{l \in [1, n_{M_i}]} z^m_{i,l} \leq 0 \tag{6}$$

$$\forall i \in \mathbf{N}, l \in [1, n_{M_i}] : z^m_{i,l} - z^c_i \leq 0 \tag{7}$$

There is a symmetry when two ports are connected, when one is switched off then the connected one is also switched off:

$$\forall i, j \in \mathbf{N} : z^e_{i,j} - z^e_{j,i} = 0 \tag{8}$$

2.2 Application Model

Variables and functions for the applications We consider one-to-one applications that are composed of one sender and one receiver. This will simplify the traffic matrix, the final notation and the number of unknown values. Yet the generalization of more complex applications is possible for the model used. The following variables are defined:

- A is the set of applications on the datacenter:
  $\mathbf{A} = \{a_1, \dots a_k \dots, a_{n_A}\}, \mid \mathbf{A} \mid = n_A$
- a_k^s is the sender process of the application a_k and a_k^r the receiver (generalisation with several receiver is possible). We suppose that the communication is one-way and we neglected back traffic (signaling, acknowledgements, etc)
- N_s^k is the host of a_k^s and N_r^k the host of a_k^r
- **NE** is the set of QoS needs for all applications. Every element of this set is composed of a set of values expressing an elementary QoS characteristic asked by the application (flow, jit, response time, ...) which are grouped together in a tuple. **NE** is composed of numerical values, interval or other representations allowing to characterize an elementary need in QoS $\mathbf{NE} = \{n_1, \dots n_s \dots, n_{n_{NE}}\}, \mid \mathbf{NE} \mid = n_{NE}$
- $\mathbf{NE^k}$ allows to specify for an application a_k the various possible needs for this application.
- $\mathbf{B^k}$ is a set which has the same size of $\mathbf{NE^k}$:
  $\mathbf{B^k} = \{b_1^k, \dots b_n^k \dots, b_{n_{B^k}}^k\}, \mid \mathbf{B^k} \mid = n_B^k$. It is composed of binary variables $b_n^k \in \mathbf{B}^k$:
$$b_n^k = \begin{cases} 1 \text{ if the need } n_s^k (s = n) \text{ is chosen for the} \\ \quad \text{application } a_k, \\ 0 \text{ otherwise} \end{cases}$$
- We suppose that there is a metric function called M allowing to measure the quality of a need. This last one allows to define a total order relation noted $<$ between the various needs of an application. To simplify afterward the notation, we suppose that needs are altogether tidied up $\mathbf{NE^k}$ following this order $<$ using the metric M (this is always achievable with an index permutation):
$$M(n_1^k) < M(n_2^k) < \dots < M(n_{n_{NE^k}}^k)$$
- $x_{i,j}^k$ is the network data flow for the application a_k on the edge (i, j)
- AF gives for an application a_k and a chosen need n_m^k, the average flow produced by this application. Our approach supposes that the average flow produced by the application according to its needs can be estimated.
- The function RoutingNodes for the network takes a sender node and a receiver node and creates the set of nodes used to go from the sender node to the receiver node depending on the routing policy (OSPF, RIP, RIPv2, etc). This function takes into account the unusable switched off nodes. For an application a_k:
$$RN^k = \text{RoutingNodes}(N_s^k, N_r^k)$$

– for a node i, the set $\mathbf{RN^{k,i}_{input}}$ defines the set of nodes connected to node i and sending data for the application a_k and $\mathbf{RN^{k,i}_{output}}$ the set of nodes connected to node i and receiving data for the application a_k:
$j \in \mathbf{RN^{k,i}_{input}}$ if $j \in RN^k$ and $\exists e_{i,j} \in \mathbf{E}$
$j \in \mathbf{RN^{k,i}_{output}}$ if $j \in RN^k$ and $\exists e_{j,i} \in \mathbf{E}$

Relations and constraints for the applications For an application, only one single need can be chosen:

$$\forall k \in [1, n_A] : \sum_{n=1}^{n_{B^k}} b_n^k - 1 = 0 \tag{9}$$

The flow going out of the sending node is equal to the average flow produced by the application and follows the routing policy :

$$\begin{aligned} &\forall k \in [1, n_A], j \in \mathbf{E} \text{ so that } \exists e_{N_s^k,j} \in \mathbf{E} : \\ &x^k_{N_s^k,j} - \sum_{n=1}^{n_{B^k}} (AF(n_n^k).b_n^k) = 0 \end{aligned} \tag{10}$$

For an application a_k the flow which goes out of the sending node is equal to the flow which goes in the receiving node. We suppose that hosts are connected to border routers with only one link:

$$\begin{aligned} &\forall k \in [1, n_A], \forall i \in \mathbf{E} \text{ so that } \exists e_{N_s^k,i} \in \mathbf{E}, \\ &\forall l \in \mathbf{E} \text{ so that } \exists e_{l,N_r^k} \in \mathbf{E} : \\ &x^k_{N_s^k,i} - x^k_{l,N_r^k} = 0 \end{aligned} \tag{11}$$

The conservation of the flows across the core network is expressed as follow. There is no lost of information, and all information that enter in a core router should exit :

$$\forall k \in [1, n_A], \forall i \in \mathbf{E} : \sum_{\substack{j \in \mathbf{E}, \exists e_{i,j} \\ j \neq N_s^k, N_r^k}} x^k_{i,j} - \sum_{\substack{u \in \mathbf{E}, \exists e_{u,i} \\ u \neq N_s^k, N_r^k}} x^k_{u,i} = 0 \tag{12}$$

The capacity of the switched on links must be respected:

$$\forall (i,j) \in \mathbf{E} : \sum_{k=1}^{n_A} (x^k_{i,j} - c^e_{i,j}.z^e_{i,j}) \leq 0 \tag{13}$$

$$\forall (i,j) \in \mathbf{E}, \forall k \in [1, n_A] : -x^k_{i,j} \leq 0 \tag{14}$$

We define a quality loss of service QL for the application a_k with the chosen quality need of service c as being:

$$\forall k \in [1, n_A] : QL_{a_k} = M(n^k_{n_{NE_k}}) - \sum_{c=1}^{n_{B^k}} M(n_c^k).b_c^k$$

The total quality loss for all applications on the datacenter is:

$$QL_{Total} = \sum_{a_k \in \mathbf{A}} QL_{a_k}$$

2.3 The Routing Protocol

Two cases have to be discussed. In the first one, the routing policy is not constrained (called optimal policy). All routes can be used and the optimal solution represents the optimal propagation flow on the network. In the second one, the OSPF policy is used. In that case, it is necessary to add constraints which specify that data flows are fairly divided on the routes having the same cost (the cost being calculated by the OSPF-specific metrics, usually the links capacities):

$$\begin{aligned} &\forall k \in [1, n_A]; \forall i \in \mathbf{RN^k}, Cardinal(\mathbf{RN^{k,i}_{output}}) > 1, \\ &i \neq N_s^k; \forall j \in \mathbf{RN^{k,i}_{output}} : \\ &x_{i,j}^k = \frac{1}{Cardinal(\mathbf{RN^{k,i}_{output}})} \sum_{l \in \mathbf{RN^{k,i}_{input}}} x_{l,i}^k \end{aligned} \quad (15)$$

2.4 Global Criterion

The problem can be written as the minimization of a criterion, by going through the possibilities for the various variables $z = [0;1]$ (the switched on/off elements of the managed network), $b = [0;1]$ (the chosen QoS needs for the applications) and $x \in \mathbb{R}^+$ (the flow values of the network):

$$\underset{x \in \mathbb{R}^+; z=[0;1]; b=[0;1]}{\text{Min}} \alpha . P_{total} + \beta . (1-\alpha) . QL_{total} \quad (16)$$

We introduce $\alpha \in [0;1]$ which represents the compromise between the total power of the managed network and the service quality loss. It is the duty of the datacenter's administrator to define α.

β allows a normalization to return both criteria on a comparable scale. This is made using the minimal and maximal borders of P_{Total} and QL_{Total}. These borders can be easily calculated for P_{max} and QL_{max} by setting all the power variables and quality of service variable to the maximum. To calculate P_{min} and QL_{min} we use a pre-optimization by setting $\alpha = 0$ and $\alpha = 1$. β is calculated by means of an average arithmetic on these borders:

$$\beta = (P_{min} + P_{max}) / (QL_{min} + QL_{max}) \quad (17)$$

3 Experiments

3.1 Context

Resolution context To validate our approach, we chose to use the Grid'MIP topology (a part of the grid'5000 project [13]) described by Fig. 1. The three

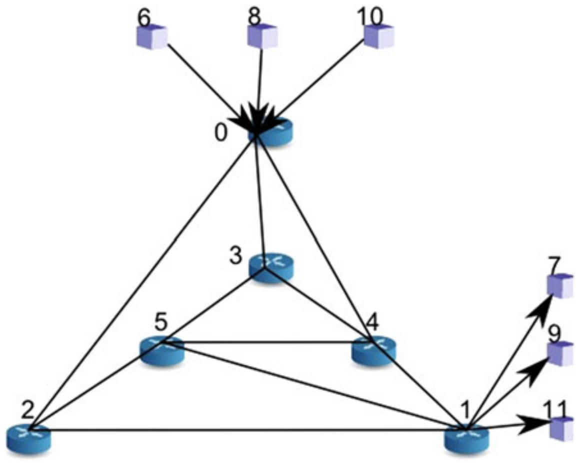


Fig. 1. Topology of the Grid'MIP architecture

border routers are numbered 0, 1, 2 and have two modules: one has 48 Gigabit-ethernet ports (each link connected has a capacity of $c_{i,j} = 1000$ Mbits/s) and the other has four Fiber Channel 10-Gigabit ports ($c_{i,j} = 10000$ Mbits/s). The first module is linked to hosts that run applications and the second module to core routers. The three core routers are numbered 3, 4, 5 and have one Fiber Channel module with four 10-Gigabit ports. For the initialization of the power constants ($p_i^c, p_{i,l}^m, p_{i,j}^e$) we used values measured on the Grid'MIP platform using specific bluetooth wattmeters called Plogg.

Regarding the application placement, we decided to use a simple layout that allows to highlight the switching on/off difference of the network equipments in two precise cases: the optimal-policy case and the OSPF-policy case. That's why we chose to place all the sending applications on hosts linked to the border router 0 and all the receiving applications on hosts linked to the border router 1 ($\forall k \in [1, n_A] : N_s^k = 0, N_s^k = 1$). Thus, the host 6 sends traffic to host 7, host 8 to host 9, etc. If we consider the optimal case there are 3 possible routes from router 0 to router 1: 0-2-1, 0-4-1 and 0-3-5-1. All links of these three routers being different and having a 10-Gbit/s capacity, the total possible bandwidth from router 0 to router 1 is 30 Gbit/s. If we then consider the OSPF case, only 2 routes remain because of the OSPF metric based on capacity: 0-2-1 and 0-4-1. The total bandwidth is therefore 20 Gbit/s. For the average flow constants $AF(n_n^k)$ we consider that all applications are consistent for a better result readability. We associate 5 basic QoS needs to these applications ($|\ \mathbf{B^k}\ | = n_{B^k} = 5$) and for simplification purposes we consider the metric M defining the quality of each need to be equal to the average flow resulting from the chosen need, i.e. $M(n_n^k) = AF(n_n^k)$. The 5 average flows resulting of the 5 needs are fixed in Mbits/s to $AF(n_0^k) = 200$, $AF(n_1^k) = 400$, $AF(n_2^k) = 600$, $AF(n_3^k) = 800$ and $AF(n_4^k) = 1000$. Finally, for our experiments, we vary the number of applications between 1 to 60 ($n_A \in [1..60]$) and α by steps of 0.5.

3.2 Simple Configuration

Resolution for the optimal case The variables of the global model are never multiplied together, so this is a linear programming (**LP**) problem. However, it is necessary to use discrete variables when modeling the problem, for example for the values associated with binary variables of the problem. In this particular case, the model adds integrity constraints and the problem is known as integer linear (**LP**) programming.

For the first optimal calculation we use JOpt [14], an open-source java tool that encapsulates **LP**. JOpt adds a generic layer to linear solvers by using java objects. This allows to access distant solvers like CPLEX [15]. JOpt manages the calculation with an internal load-balancing policy over multiple solvers.

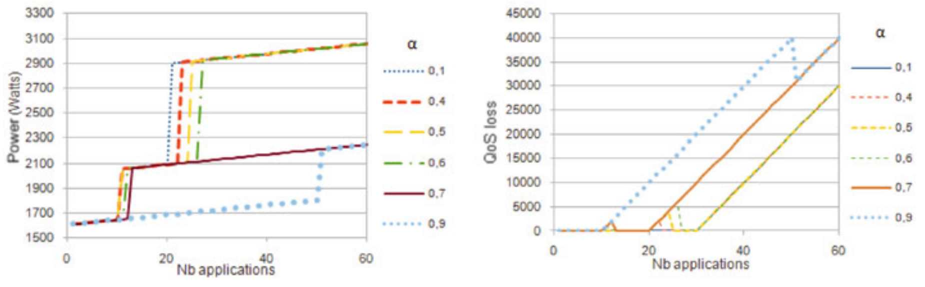
For the criterion resolution, four steps are needed for the optimal case:

– **Step 1 Initialization:** The first step initializes the problem inputs: network graph (routers, modules, ports and links), the placement of the applications on the graph (for each a_k creation of the sending and receipting nodes, of the application needs and calculation of the average flows) and the constants.
– **Step 2 Preparation:** The second step prepares the criterion for the final objective function and the constraints. In fact, this step calculates the coefficients and constructs the JOpt java objects: variables, terms, criteria, constraints and objective function.
– **Step 3 Normalization:** The third step allows to calculate the normalization needed for the objective function. Two calls on the distant solver are needed for this step for the calculation of P_{min} and QL_{min}. P_{max} and QL_{max} are also calculated but do not need a solver.
– **Step 4 Minimization:** The forth step consists in launching the minimization of the distant solver and getting the results back.

The calculation of the coefficients and the construction of the JOpt java objects (variables, terms, criteria, constraints and objective function) are being made on a JOpt client coded in java that transfers these objects to the distant linear solver. In our case, we use a CPLEX solver on a distant server with four processors dual-core Intel Xeon 3.2 GHz. In our experiments we distinguish between the **preparation_time** needed to prepare the calculation and the **network_time** needed to transfer the JOpt java objects and get the results back.

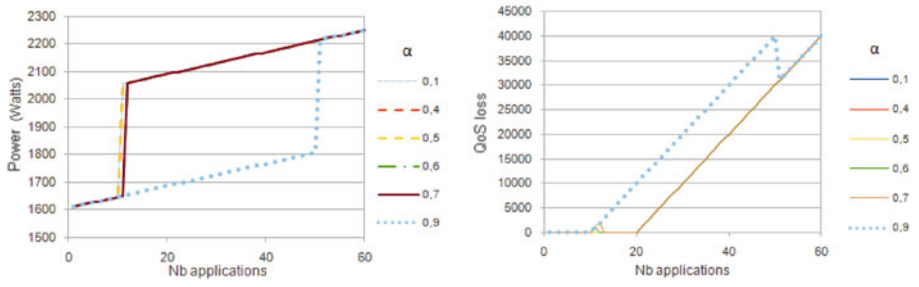
Figures 2a and 3a show the electric power needed by the network as a function of the number of applications and some selected α values.

In the optimal case, (Fig. 2a), between 0 and 10 applications for $\alpha < 0.8$ there is only one route switched on (on Fig. 1 the route 0-2-1 or 0-4-1) and the power consumption increases slowly from 1600 to 1700 Watts as a function of the number of ports (approximately 4 ports per application: 2 for the sender and 2 for the receiver, so 4 Watts per application). For $\alpha = 0,9$ the consumption increases following this scheme until 50 applications. Between 11 and 13 applications for $\alpha < 0,8$, we can see a jump in the power consumption indicating that a router (frame, modules and ports concerned) is switched on. We have to wait until 50



(a) Power consumption (b) Quality loss of service

Fig. 2. Optimal self-optimization case



(a) Power consumption (b) Quality loss of service

Fig. 3. Heuristic with OSPF routing protocol for self-optimization case

applications to observe this jump for $\alpha = 0,9$. Indeed, after 50 applications, the average flow resulting from the minimum needs is fixed to 200 Mbits/s and the capacity of this unique route (10 Gbit/s) is exceeded which forces the switch on of a second route.

We observe that the more α increases the more power is taken into account in the minimization, delaying the switch on of new routers to the detriment of quality of service, as can be seen on Fig. 2b. Between this first jump and the second after 20 applications we have routers 0, 1, 2 and 4 switched on and the routes 0-2-1 and 0-4-1 used. Starting at 20 applications and for $\alpha < 0,7$, we see a second power jump more important than the first one because it concerns the switch on of routers 3 and 5. This jump is moving from 21 applications for $\alpha = 0,1$ to 27 applications for $\alpha = 0,6$. We see experimentally that for $\alpha = 0,7$ this jump isn't happening (until 60 applications). With the routers 3 and 5 switched on, the network reaches a power consumption of more than 2900 Watts and 3 routes are used (0-4-1, 0-2-1 and 0-3-5-1).

Regarding the QoS (Figs. 2b and 3b), we see that globally the more α is increasing, the more quality loss of service the user gets. We also see that the quality losses coincide with the power jumps of Figures 2a and 3a. For example for $\alpha = 0,9$, the power is predominant in the objective function. In this case we see that the quality loss of service starts after 11 applications. Indeed, until 10 applications, the total average flow cannot exceed the capacity of the only route switched on (either 0-2-1 or 0-4-1). Thus these 10 applications use the needs whose quality measured by M is maximum and have a resulting average flow $AF(n_4^k) = 1000$ Mbits/s. After 11 applications, the QoS is degraded for at least one application. This phenomenon is visible just before new routers are switched on. We can indeed observe a slight increase of quality loss, visible from 20 to 27 applications for $\alpha < 0,7$ in the optimal case (Fig. 2b).

Algorithm 1 Heuristic used with OSPF as the routing function

```
Initialize constants, constraints and execute algorithm OSPF
Solve $P_{min}$ (LP with $\alpha = 1$), $QL_{min}$ (LP with $\alpha = 0$)
Solve directly $P_{max}$, $QL_{max}$
$\beta = (P_{min} + P_{max})/(QL_{min} + QL_{max})$
Create Solution best_solution ← Solve LP
Create List explored_solutions ← best_solution
while $nb\_iterations < max\_iter$ & $moving\_objective$ do
    $\mathbf{N^{trie}}$ ← sort $\mathbf{N}$ by inverse number of applications $a_k$ using them
    for $i \in \mathbf{N^{trie}}$ do
        $\mathbf{M_i^{trie}}$ ← sort $\mathbf{M_i}$ by inverse number of applications $a_k$ using them
        for $l \in \mathbf{M_i^{trie}}$ do
            $\mathbf{E_{i,l}^{trie}}$ ← sort $\mathbf{E_{i,l}}$ by inverse number of applications $a_k$ using them
            for $e_{i,j} \in \mathbf{E_{i,l}^{trie}}$ do
                execute algorithm 2 with $z_{i,j}^e = 0$
                $nb\_iterations \leftarrow nb\_iterations + 1$
            end for
            execute algorithm 2 with $z_{i,l}^m = 0$
            $nb\_iterations \leftarrow nb\_iterations + 1$
        end for
        execute algorithm 2 with $z_i^c = 0$
        $nb\_iterations \leftarrow nb\_iterations + 1$
    end for
end while
return best_solution
```

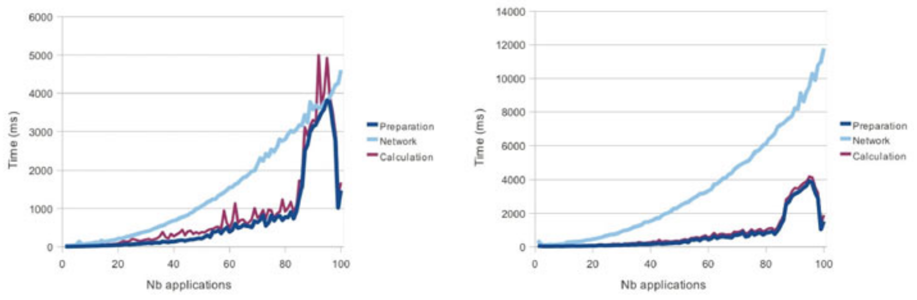
Resolution when using a heuristic The use of a heuristic when the network routing policy is OSPF is mandatory because when minimizing the global objective function, for each change of the value of variable z, the routing function changes. Because the **Constraint** 15 becomes dependent on a variable to minimize, we introduce a heuristic proposed by algorithm 1. The idea consists in calculating a first solution that uses all OSPF routes for all applications. This is done at the beginning of the heuristic with "algorithm OSPF". This algorithm adds constraints that force the flows using non-OSPF routes to be null. Before starting the iteration loop of the heuristic, a first solution (saved in variable

Algorithm 2 Verifying function for the heuristic

```
Require: $z$ as input; explored_solutions & best_solution as input/output
  if $z = 0 \notin$ explored_solutions then
    if $\forall k \in [1, nA]$ & $z = 0$ & $\exists OSPF\_route(N_s^k, N_r^k)$ then
      Solution s ← execute algorithm OSPF and Solve LP with best_solution and
      $z = 0$
      if ∃s then
        explored_solutions ← s
        if s is best then
          best_solutions ← s
          $nb\_tries \leftarrow 0$
        else if $nb\_tries < max\_tries$ then
          $nb\_tries \leftarrow nb\_tries + 1$
          explored_solutions ← s
        else
          $moving\_objective =$ false
        end if
      else
        explored_solutions ← s
        stop the for loop
      end if
    else
      explored_solutions ← s
      stop the for loop
    end if
  end if
```

best_solution) is calculated taking into account these constraints. A history saved in variable **explored_solutions** allows to memorize an association between the set of solutions for the variables and the set of constraints added by the heuristic to avoid recalculating a solution with a topology that has already been explored.

The heuristic is stopped if the maximum number of iterations is reached ($nb\_iterations < max\_iter$) or if the best solution hasn't changed since a number of iterations $moving\_objective$). We sort the frames (nodes), the modules and the ports by number of flows using them (smaller number first). We try to switch off first the ports then the modules and finally the frames (nodes) using the sorted list. Indeed, this minimizes the number of applications impacted when switching off the equipment. For each switching off we verify if one OSPF route still exists for all applications and if one solution exists using algorithm 2. This algorithm is the same for the ports ($z_{i,j}^e = 0$), the modules ($z_{i,l}^m = 0$) or the frames (nodes) ($z_i^c = 0$) and we describe it in a generalized way with variable z. If this algorithm finds an OSPF route for all applications and a better solution, it is saved in **best_solution**, otherwise it continues exploring the solutions for max_tries iterations. Every time a better solution is found $nb\_tries = 0$. If we



(a) optimal case (b) heuristic case

Fig. 4. Resolution time

reach *max_tries* without finding a better solution then we stop the heuristic *moving_objective* = **false**.

When using the heuristic (Figs. 3a and b) we observe that the results follow the optimal ones but never switch on routers 3 and 5. As explained in Sect. 3.1, when using an OSPF-policy, the route 0-3-5-1 cannot be used for load balancing. Indeed, when using the link capacities as the OSPF metric the cost of route 0-3-5-1 is more important and will never be used for the routes from router 0 to router 1. Regarding the QoS, it is therefore globally more degraded than in the optimal case because the total bandwidth is lower without this third route.

3.3 Comparison of the Optimal and Heuristic Cases

Figures 4a and b show the resolution times of the objective function in the optimal and heuristic cases. We varied the number of applications to 100 to show the impact on the different times:

– Preparation time: calculation of the objective, variables, terms and JOpt constraints(developed form).
– Network time: transfer time for the set of optimization parameters to the CPLEX server and transfer time to get the results back.
– Calculation time: the raw and only calculation time for the criterion by the CPLEX server.

Each resolution makes three calls to the distant CPLEX server. Indeed, two calls are needed for the calculation of β (for P_{min} and QL_{min}) and a call for the resolution of the criterion. The preparation and calculation times include the times for P_{min}, P_{max}, QL_{min}, QL_{max} and the final problem. The network transfer times include the three distant CPLEX calls.

We observe that the preparation and calculation time are substantially the same in the optimal and heuristic cases. These times vary between 20 ms for one application to one second for 80 applications. After 80 applications, these times increase to 4 seconds because it is more difficult to find a solution because all network links considered in the experiment are saturated. For 100 applications there is no possible solution so the times decrease to one second.

Regarding the network transfer times for the JOpt data to the CPLEX server, we see that when we use the heuristic it is more important than in the optimal case. Indeed the heuristic adds additional constraints related to the routing policy. Adding constraints increases the number of variables and terms to be transmitted to the server for the resolution. In the heuristic case, this time varies exponentially from 360 ms for one application to 12 seconds for 100 applications.

Given the results, we can conclude that using a heuristic doesn't generate an overhead for the calculation time or preparation time. However, the network transfer time is multiplied by 2.6. Globally, these times are still reasonable in the case of dynamic reconfiguration of network devices like routers. Therefore, the optimization of the network has to be planned with a granularity of about an hour, which makes the resolution time negligible.

4 Conclusion

Human administrators cannot face the complexity of management of the IT infrastructure and the deployed application on datacenters anymore. Whether it concerns hardware or software issues, the optimization process is a tricky and costly task. We introduced the "self-optimizing" autonomic property at the hardware level by applying it to the optimization of datacenters energy costs.

We introduced an approach to describe a compromise between, on one hand, the power consumption of the network infrastructure and on the other hand, the deterioration of the QoS for the applications using this network. Being able to control the dynamic reconfiguration at two levels: at the application level by dynamically reconfiguring QoS profiles and at the hardware level by switching on and off links, modules or routers allows to have a global management of the datacenter. Indeed, the use of an autonomic manager allows the administrator to control the energy costs or the performance by varying only one parameter that handles a high level management policy.

Regarding the limits of our approach, we suppose that a relation of order exists between the QoS needs by using the function M. This order is not to be confused with the final user "quality of experience" (QoE) [16].

The goal was to deal with the performance/electric consumption dilemma for the network part. This is a challenge for years to come as the perfect system must take into account the energy consumption of the machines and also the network equipments, the QoS and financial costs for example.

Acknowledgment. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804. Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

References

1. Skene, J., Lamanna, D.D., Emmerich, W.: Precise service level agreements. In: 26th International Conference on Software Engineering (ICSE'04), Edinburgh, Scotland, United Kingdom (2004)
2. Chang, F., Karamcheti, V.: Automatic configuration and run-time adaptation of distributed applications. In: High Performance Data, Computing, p. 11 (2000)
3. Grossman, D.: New terminology and clarifications for diffserv. Technical report, RFC 3260 (2002)
4. Hsueh, C., Lin, H., Huang, G.C.: Channel bonding in linux ethernet environment using regular switching hub. Syst. Cybern. Inform. **2**(3), 35–38 (2004)
5. Callon, R.W.: Use of OSI IS-IS for routing in TCP/IP and dual environments. Technical report, RFC 1195 (1990)
6. Moy, J.: Ospf version 2. Technical report, RFC 2328 (1998)
7. Zinin, A.: Cisco IP routing: packet forwarding and intra-domain routing protocols. Addison-Wesley, Boston (2002)
8. Albrightson, R., Garcia-Luna-Aceves, J.J., Boyle, J.: EIGRP-a fast routing protocol based on distance vectors. In: Proceedings of the Networld/Interop, vol. 94 (1994)
9. Liu, X., Ghazisaidi, N., Ivanescu, L., Kang, R., Maier, M.: On the tradeoff between energy saving and qos support for video delivery in eee-based fiwi networks using real world traffic traces. Lightwave Technol. J. **29**(18), 2670–2676 (2011)
10. Arai, D., Yoshihara, K.: Eco-friendly distributed routing protocol for reducing network energy consumption. In: International Conference on Network and, Service Management, October 2010
11. Chabarek, J., Sommers, J., Barford, P., Estan, C., Tsiang, D., Wrigh, S.: Power awareness in network design and routing. In: Proceedings of the IEEE INFOCOM (2008)
12. Dong, X.W., El-Gorashi, T., Elmirghani, J.M.H.: Green ip over wdm networks with data centers. Lightwave Technol. J. **29**(12), 1861–1880 (2011)
13. Capello, F., Caron, E., Dayde, M., Jegou, Y., Desprez. F, Primet, P., Jeannot, E., Lanteri, S., Leduc, J., Melab, N.: Grid'5000: a large scale and highly reconfigurable grid experimental testbed. In: 6th IEEE/ACM International Workshop on Grid Computing, Seattle, Washington, USA, pp. 99–106 (2005)
14. Shneidman, J.: JOpt, a simplified java wrapper for linear and mixed integer programming. Technical report, http://www.eecs.harvard.edu/econcs/jopt/ (2005)
15. IBM: Mathematical programs - IBM ILOG CPLEX optimizer - software. Technical report. http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/ (2009)
16. Jain, R.: Quality of experience. IEEE Multimed. **11**(1), 96–97 (2004)

Large Scale Analysis of BitTorrent Proxy for Green Internet File Sharing

Sena Cebeci[1(✉)], Oznur Ozkasap[1], and Giuseppe Anastasi[2]

[1] Department of Computer Engineering, Koc University, Istanbul, Turkey
{senacebeci, oozkasap}@ku.edu.tr

[2] Department of Information Engineering, University of Pisa, Pisa, Italy
giuseppe.anastasi@iet.unipi.it

Abstract. Addressing energy efficiency in P2P services has the potential to make the Internet greener since they comprise a major source of the Internet data traffic. In this paper, we consider approaches for green Internet file sharing. We develop simulation models of proxy-based energy efficient BitTorrent as well as legacy BitTorrent on PeerSim P2P simulator, and explore their characteristics on large-scale scenarios. Our aim is first to identify the operating conditions where proxy-based BitTorrent outperforms the legacy protocol in terms of overall energy efficiency, and then to develop optimizations for the proxy-based approach.

Keywords: BitTorrent · Energy efficiency · Green Internet

1 Introduction

With energy efficiency becoming a high-priority concern in today's world, design and development of energy-aware ICT services attract increasing attention. Recent studies have shown that the internet related energy consumption represents a significant part of the overall energy consumption of ICT because Internet traffic continues to increase rapidly [1]. Moreover, P2P traffic due to file sharing represents a large fraction of the Internet traffic. Thus, P2P services and protocols largely contribute to the Internet related energy consumption. Therefore, it becomes more and more important that the P2P protocols need to consider energy efficient approaches to support green communication and content sharing.

Among the P2P file sharing protocols, BitTorrent is the highly popular protocol since BitTorrent P2P traffic constitute more than 45-78% of all P2P Internet traffic, and roughly 27-55% of all the Internet traffic depending on geographical location [2]. Solutions for green file sharing approaches are broken down into three categories, and interpreted according to the power management technique they used [3]. These categories are adaptive link rate, different power management levels, and proxy-based. In adaptive link rate and power management levels energy efficiency is dependent on the hardware performance of the NIC. These techniques are not good candidates for internet file sharing, thus we have chosen proxy-based solution. In proxy-based approach file download is fully performed by the proxy, by switching off the requested user's PC in course of the download.

DOI: 10.1007/978-3-642-40517-4_21,

A proxy-based BitTorrent called Energy Efficient BitTorrent (EE-BT) has been proposed in [4], its comparison with the legacy BitTorrent protocol has been carried out using an experimental testbed with limited number of peers. In this work, we intend to compare the performance of these approaches in large-scale scenarios with a large number of peers, and we can control protocol parameters. With the analysis, we aim to find out under which operating conditions EE-BT outperforms the legacy protocol in terms of overall energy efficiency. Goals of our study include (1) Developing models of EE-BT and legacy BitTorrent on PeerSim P2P simulator [5], (2) Performing simulations of EE-BT and legacy BitTorrent on large-scale scenarios to investigate several parameters, (3) Developing extensions/optimizations to EE-BT such as investigating the optimal number of peers a proxy can serve without degrading the performance, and (4) Analyzing the effect of increasing the number of proxies on the overlay.

2 BitTorrent Architecture Types

In this section we briefly describe the Legacy BitTorrent protocol [7] and Energy Efficient BitTorrent protocol [4].

2.1 Legacy BitTorrent

BitTorrent is a file distribution system essentially developed to distribute files with large size. According to the protocol terminology, nodes of the overlay are called peers, and the set of peers involved in the distribution of a file is referred to as torrent or swarm. Each peer downloads the desired file, in chunks, from a multitude of other peers instead of fetching it from a single server (as in the conventional client-server model). While downloading missing chunks, peers upload to other peers in the same torrent the chunks they have already obtained. For each torrent there is a tracker, i.e., a node that constantly tracks which peers are involved in the torrent. A peer that wants to join a torrent must register with the tracker and, then, it must periodically inform the tracker that it is still in the torrent.

2.2 Energy Efficient BitTorrent (EE-BT)

EE-BT is a proxy-based version of BitTorrent where download requests of peers are served by a BitTorrent proxy. Peers that are not directly involved in the torrent are called passive peers. They delegate the task to the associated proxy, which downloads the file on behalf of them. The proxy participates to the BitTorrent overlay network, just like any other regular peer, and takes care of the overall process. Therefore, the user's PC can be switched off during the download phase. The file will be transferred from the proxy to the user's PC later, when the user reconnects. Figure 1 shows the actions performed by the various actors of EE-BT. Upon receiving a request from the user, the software running on the user's PC contacts the proxy and requests the desired file. The

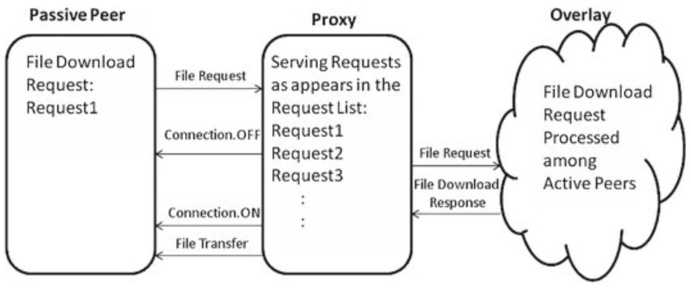


Fig. 1. EE-BT Protocol Sequence Diagram

proxy acknowledges the request with file request message. If the requested file is already available in the local cache of the proxy, it is immediately transferred to the user. Otherwise, the proxy starts downloading the requested file from the BitTorrent overlay network, acting as a regular peer and following the legacy BitTorrent protocol. The user's PC can be switched off just after receiving the acknowledgement from the proxy. Later, when the user reconnects, he/she can check the status of the download process at the proxy. If the file is completely available, it can be transferred from the proxy to the user's PC.
Other potential tasks that might be performed by the proxy can be listed as follows:

1. Proxy connects to other proxies in the network in order to serve the requests of the other peers belonging to different proxies.
2. Proxy is capable of serving a request of a peer which is not necessarily a neighbor of itself.

3 Analysis and Experimental Results

We have developed simulation models for Legacy and EE BT protocols. The simulation environment for these protocols is PeerSim. Peersim provides a dynamic environment to develop and test any kind of P2P protocol with high scalability (scales up to 1 million nodes), easy to configure and simple component-based architecture.
Scenario Description: We designed scenarios to measure average download time, average number of pieces uploaded/downloaded by peers and energy savings. In our system, network size is configurable and distribution of the seeders and leechers in the swarm has default values, equal to 20% seeders and 80% leechers, respectively. We consider the distribution of a single file in each experiment. Initially, all the peers in the system have file pieces randomly distributed, and all the peers in the network request the same file. The simulation terminates after all the peers downloaded the file successfully. These peers form the topology in the peerlist generated by the tracker and all the peers in the list are presumed to be neighbours of each other. We used three different file sizes 100 MB, 500 MB

Table 1. Default Parameter Values in the Simulation Setting

Parameter	Value
Network Size	1000
Piece Size	256 KB
Seeder Distribution	20%
Leecher Distribution	80%
File Size	100 MB
Upload/Download Rate	640-4096 Kb/s
Message MinDelay	50 ms
Message MaxDelay	200 ms

and 1 GB for our scenarios, and file pieces are 256 KB each. Upload and download data rates per peer are selected randomly from four different options, 640 Kb/s, 1 Mb/s, 2 Mb/s and 4 Mb/s, in order to use in the transmission time of a file. To set the delays, Peersim Transport Package is used. This package includes Uniform Random Transport protocol that provides an environment for reliable message delivery between peers with random delays. Default values of minDelay (minimum delay of messaging) is set to 50 ms and maxDelay (maximum delay of messaging) is set to 200 ms. Default values used in the experiments are given in Table 1.

To explore the protocol behaviours in the network, analysis on average download time, average number of pieces uploaded/downloaded and energy savings carried out. The analyzed metrics for a peer are:
Average Download Time: The duration (in milliseconds) to download the torrent file by a generic peer in the network.
Average Number of Pieces Uploaded/Downloaded: This metric measures the uploaded and downloaded pieces of the file by peer during the simulation, thus the file download percentage can be easily kept track of.
Energy Saving: We have calculated performance metric considering energy savings of the Legacy BT and EE BT protocol relative to each other. Without losing in generality, we assumed that all PCs and proxy have the same power consumption as in [4] and [6]. Under this assumption, it can be shown that the energy saving achieved when using EE BT is given by [6]; (1) where with respect to Legacy BT.
The formula of the energy savings introduced by EE BT with respect to Legacy BT, for single user i, is represented by:

$$S^{i}_{usr}(\mathrm{N}) = 1 - \frac{\sum_{i=1}^{N}(D_0(i) + 2\sum_{i=1}^{N}(D_t(i))}{\sum_{i=1}^{N} D_L(i)} \tag{1}$$

N: Number of BitTorrent peers in the network
D_0(i): Time taken by the proxy to download the file (requested by peer i) from the BitTorrent overlay.
D_t(i): Transfer time of the file from EE BT to user
D_L(i): Download time of the torrent file with Legacy BT

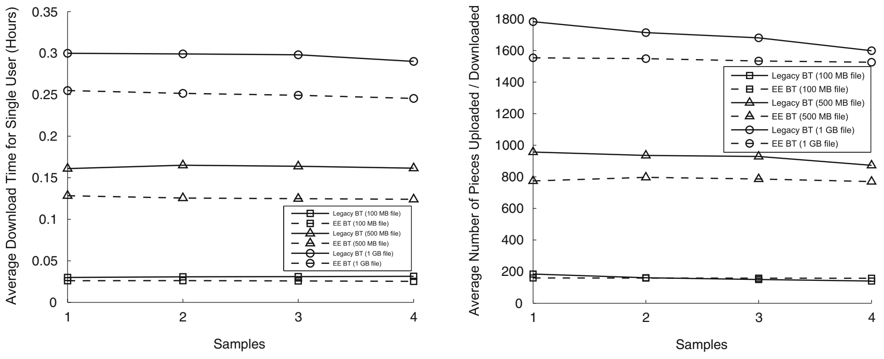


Fig. 2. Legacy BT vs EE BT (a) Average download time for single user (b) Average number of pieces downloaded/uploaded for single user

Figure 2a indicates the average download time of a torrent file having, 100 MB, 500 MB and 1 GB file sizes in legacy BT and EE BT for samples (1 to 4) of generic peers for different network sizes 10, 100, 1000, 10000, respectively. In this case, it is assumed that according to the network size all the peers either uses Legacy or Proxy-based solution for the file download. We observe that the average download times for small file sizes (i.e., 100 MB) are close to each other for both protocols. For instance, for a sample from 10 peer network size (i.e., sample 1), average download times of legacy BT and EE BT are 1.812 min (0.0302 hour) and 1.548 min (0.0258 hour), respectively. On the other hand, for large file sizes (i.e., 1 GB) the differences between these two protocols become more clear, average download rate of legacy BT is 29.8 hours whereas EE BT is 24.9 hours.

Figure 2b shows the average number of uploaded and downloaded pieces during the simulation. For the legacy BT, average upload/download rates exhibits some small differences. For example, for different samples the range of pieces differ from 184.33 to 140.10 for 100 MB file. Furthermore, this rate decreases while the number of peers increases. In other words, protocol message traffic (choke messages) affects the performance in a negative manner. On the other side, for the EE BT protocol shows average upload/download rates of pieces do not differ significantly from each other.

We have calculated Energy Savings Percentages for 100 MB and 500 MB file sizes, with upload rate 1 Mb/s. D_t(i) values for file sizes 100 MB and 500 MB are 0.22 h and 1.11 h, respectively. Average download time samples for different network sizes (i.e. Sample 1:10 peers, Sample 2:100 peers, etc.) are shown in Table 2. Energy Savings of a single user calculations in terms of percentages based on Eq. 1 are represented in Table 3.

We also consider another (hybrid) scenario when the peers in the overlay are defined as passive and active. Peers connected to the proxy for file download request are identified as passive. The remaining peers in the system are called active. In Fig. 3a, we investigate how average download time is affected by two

Table 2. Average Download Times (Single User)

Samples	D_L (100 MB)	D_{EEBT}(100 MB)	D_L(500 MB)	D_{EEBT}(500 MB)
1	0.0302 h	0.0258 h	0.1632 h	0.1123 h
2	0.0310 h	0.0260 h	0.1680 h	0.1118 h
3	0.0313 h	0.0252 h	0.1711 h	0.1112 h
4	0.0318 h	0.0247 h	0.1746 h	0.1108 h

Table 3. Single User Energy Savings

Samples	100 MB File	500 MB File
1	15.42%	14.29%
2	15.03%	13.87%
3	14.86%	13.62%
4	14.61%	13.34%

protocols if the percentage of passive peers in the network increases. The results show EE BT average download time is less than Legacy BT. For example, for a file size 500 MB EE BT download time is 13.75% better. Furthermore, for 1 GB file size EE BT average download time is 16.25% lower. As shown in Fig. 3a, for both protocols, the average download time slightly decreases when the percentage of passive peers increases.

In Fig. 3b, the increment on passive peers in the network causes relatively large decreases on average number of pieces uploaded/downloaded per peer. The significant reduction of number of pieces uploaded/downloaded is the reason of passive peers do not upload but download from the proxy.

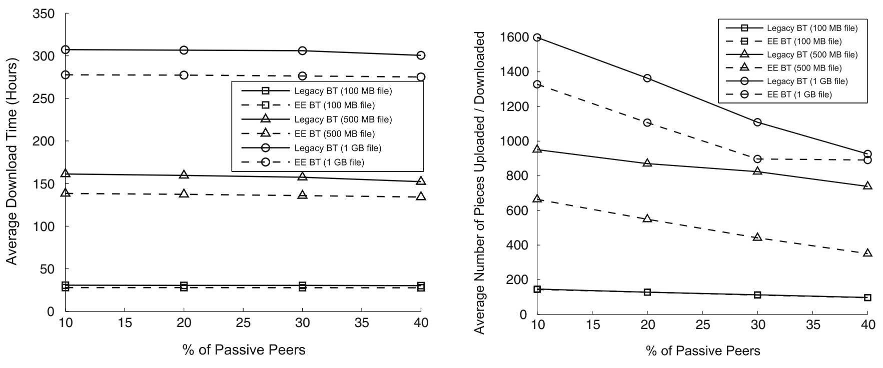


Fig. 3. Passive node percentage in Legacy BT and EE BT (a) Average download time per peer (b) Average number of pieces downloaded/uploaded per peer. The total number of peer is constant and equal to 1000.

4 Future Work and Conclusions

In this work, we have developed Legacy BitTorrent and EE BitTorrent implementations on Peersim P2P simulation environment. We have evaluated the simulation results in large-scale scenarios (i.e. with large number of peers) using three performance metrics. The first metric represents the average download time taken by the two protocols and the other one investigates average number of pieces uploaded/downloaded per peer in the run time. Lastly, user energy savings comparison measured by performance metric. We have measured the energy saving introduced by EE BitTorrent with respect to legacy BitTorrent. Our simulation results have revealed that proxy based EE BT decreases the average download time of the torrent file by 16.25%.

For future work, we plan to extend our simulation model increasing the number of proxies in the network, and in order to explore how it affects the system in terms of energy efficiency. Additionally, we will investigate optimizations and extension methods to EE BitTorrent to examine parameters such as optimal number of peers a proxy can serve without degrading the performance.

Acknowledgments. Our work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804: Energy efficiency in large scale distributed systems, and by TUBITAK (The Scientific and Technical Research Council of Turkey) under Grant 109M761.

References

1. Audzevich, Y., Moore, A., Rice, A., Sohan, R., Timotheou, S., Crowcroft, J., Akoush, S., Hopper, A., Wonfor, A., Wang, H., Penty, R., White, I., Dong, X., El-Gorashi, T., Elmirghani, J.: Intelligent energy aware networks. In: Handbook of Energy-Aware and Green Computing. Chapman and Hall/CRC, New York (2012)
2. Lee, J.Y., Jeong, J.-H., Kim, H. Y., Lee, C. H.: Energy-saving set top box enhancement in bittorrent networks. In: 2010 IEEE Network Operations and Management Symposium (NOMS), pp. 809–812 (2010)
3. Anastasi, G., Conti, M., Passarella, A.: Power management in mobile and pervasive computing systems. In: Algorithms and Protocols for Wireless and Mobile Networks (2005)
4. Anastasi, G., Giannetti, I., Passarella, A.: A bitTorrent proxy for green internet file sharing: design and experimental evaluation. Comput. Commun. **33**(7), 794–802 (2010)
5. The PeerSim Simulator. http://peersim.sf.net (2013)
6. Giannetti, I., Anastasi, G., Conti, M.: Energy-efficient P2P file sharing for residential BitTorrent users. In: IEEE Symposium on Computers and Communications (ISCC), pp. 524–529 (2012)
7. Guo, L., Chen, S., Xiao, Z., Tan, E., Ding, X., Zhang, X.: Measurements, analysis, and modeling of BitTorrent-like systems. In: 5th ACM SIGCOMM Conference on Internet Measurement (IMC '05). USENIX Association, Berkeley (2005)

Energy Efficiency Issues in Information-Centric Networking

Torsten Braun[1(✉)] and Tuan Anh Trinh[2]

[1]University of Bern, Bern, Switzerland
braun@iam.unibe.ch
[2]Budapest University of Technology and Economics, Budapest, Hungary
trinh@tmit.bme.hu

Abstract. In this paper we address energy efficiency issues of Information Centric Networking (ICN) architectures. In the proposed framework, we investigate the impact of ICN architectures on energy consumption of networking hardware devices and compare them with the energy consumption of other content dissemination methods. In particular, we investigate the consequences of caching in ICN from the energy efficiency perspective, taking into account the energy consumption of different hardware components in the ICN architectures. Based on the results of the analysis, we address the practical issues regarding the possible deployment and evolution of ICN from an energy-efficiency perspective. Finally, we summarize our findings and discuss the outlook/future perspectives on the energy efficiency of Information-Centric Networks.

Keywords: Energy efficiency · Information-centric networking

1 Introduction

The Information-Centric Networking (ICN) paradigm has shown great potential in empowering the users in future networks as well as supporting new and emerging applications. Recently, the paradigm of ICN has been developed to support future Internet applications. ICN supports the content- and service-driven communication scheme on which computing infrastructures (e.g. cloud computing) are based on. Information-Centric Networking is based on caching data in network elements, which are extended by appropriate memory to implement large caches. Caching data in network elements allows reducing the delay when accessing the data multiple times and distributes data somehow automatically without explicitly triggering the movement of data. This should improve the QoE experienced by users when accessing their data in the networks.

COST804 final workshop. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

It can be observed that much research work so far focuses on architectural issues including naming and addressing as well as transport, caching, error control, flow control in ICN. Socio-economic issues of ICN such as security, privacy, and new business models have also been considered. However, caching of data in network elements raises issues related to energy consumption as well. Despite this fact, energy issues in ICN have not been received much attention and has not been investigated thoroughly. In this paper, we address energy efficiency issues of ICN architectures. In the proposed framework, we investigate the impact of ICN architectures on energy consumption of networking hardware devices and compare them with the energy consumption of other content dissemination methods.

2 Investigation of Caching Strategies in ICN from an Energy Efficiency Perspective

Fast memory to be used for the implementation of caches is expected to consume much more energy (e.g., for refreshing cycles) than secondary memory technologies such as solid state disks (SSDs) or hard disk drives. Since caching can only be beneficial for high cache hit rates, large memories in network elements should be used. This further increases energy consumption. To limit energy consumption by ICN, appropriate mechanisms must be developed. Among those might be smart caching strategies to optimally exploit cache memories. Moreover, appropriate transport mechanisms supporting energy-efficient data transfer between cloud services/storage and mobile devices must developed and/or selected. Finally, ICN operation on top of wireless networks should use as little radio resources as possible and allow end systems to enter power-saving states as frequently as possible. Energy-saving mechanisms, however, come at the cost of quality degradation, in particular caused by lower throughput and increased delays. Energy-saving mechanisms should, therefore, be designed with QoE required by the user in mind.

2.1 Impact of Caching/Replication Strategies in ICN

Assuming a layered network topology as depicted in Fig. 1, an interesting question with ICN - from the energy efficiency perspective - is where (at which layer) to cache the content. The higher the layer at which content is cached, the less is the duplication of data (and thus the less energy for storage of those content). On the other hand, the lower the layer the content is cached, the faster is the content transmission. We can see that there is a trade-off here. It is necessary to quantify the energy consumption of transmission links and storage in order to design and evaluate the caching / replication strategies in ICN. In addition, the nature and the characteristics of the requests and the content being requested, e.g. how frequently content objects are requested, popularity of contents, etc. should be investigated as well. We consider that this is an important research challenge to be addressed in terms of energy efficiency in ICN. The idea of caching strategies mentioned above is illustrated in Fig. 1. In this simple example, content can be cached at the lowest layer, close to the clients. In this case additional

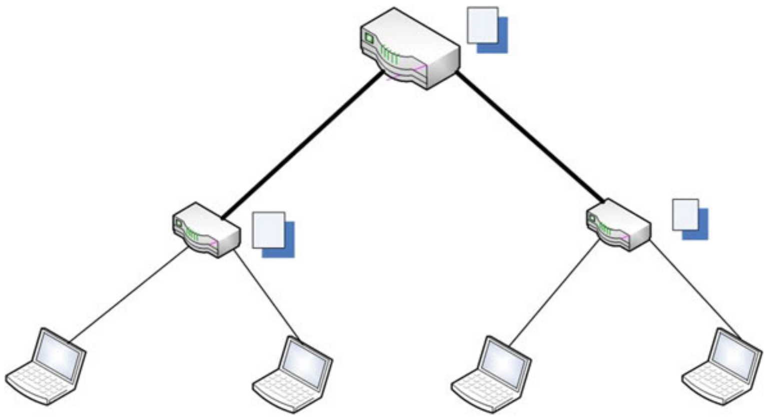


Fig. 1. Caching options

energy is needed for the storage of the duplicates of the content (2 duplicates). However, the end system has direct access to the content, only with one link (no hop in between). The other strategy is to cache the content one level higher. In this case, less energy for storage is needed, but the energy consumptions for the links are higher, because there is one hop in between (one more link needed).

Assuming that each of the clients requests the same content object, the energy costs for the case where the content is stored on the lower layer is $4T + 2S$ with T = energy costs for transporting a content object across a link, S = energy costs for storing a content object in the cache. For the case, where the content object is only stored at the highest layer, the overall energy costs are $S + 8T$. Caching on the lower layer is cheaper as long as $S < 4T$.

2.2 Hardware Requirements

From a hardware perspective, it is important to note that memory access is the main bottleneck for packet processing of today's hardware router design. Energy consumption of additional memory largely depends on the performance of the memory components. Considering [1, 6] we estimate approximately 3 W/GB for DRAMs, 0.02 W/GB for hard disks, and 0.01 W/GB on SSDs in the following. The additional energy required for memory must be compared to energy required for transmission, which is estimated in [1] as 15 W/Gbps = 15 Joules/Gigabit.

ICN such as Content-Centric Networking (CCN, [9]) requires various additional memories in access and core routers. For CCN implementations additional memory for Forwarding Information Base (FIB), Pending Interest Table (PIT), and Content Store (CS) for packet caching is needed. To support line rates of 1 Gbps fast memories with possibly increased energy consumption are needed. Perino and Varvello [5] estimates that an appropriate core router consuming 5 kW without CCN support will use more than additional 3 kW due to additional memories needed for CCN. For the investigation of CCN-supported core routers, a Cisco CRS 1 router with 8 40-Gbps line cards has been considered. The authors propose to add 10 GB CS DRAM per line card plus some RL-DRAM for index tables as well as PIT and FIB memory in the range of GB range. As a result, the analysis in [5] concludes that the additional energy consumption to extend a core router to CCN functionalities would be of 3.3 W.

With respect to edge routers, a Cisco 7505 has been considered in [5]. In this case, the authors proposed to use a 1-TByte high speed SSD for the packet store, a 6 GByte DRAM for HC-log indexing, and 200 Mbit SRAM for the on-chip FIB memory. Combined together, this configuration is expected to perform LPM (Longest Prefix Matching) on about 20 million prefixes at a maximum speed of 15 Mpackets/s. The analysis there also indicates that the additional energy consumption to extend an edge router (e.g. Cisco 7505) to CCN functionalities would be in the range of 200 W, while the original peak power is 400 W. A key lesson learnt from [5] is that it is possible and feasible to support CCN deployment in the Content Distribution Network (CDN) or Internet Service Provider (ISP) scale for reasonable additional cost and energy consumption. However, today's technology is not yet ready to support an Internet scale CCN deployment.

2.3 Analysis of Energy Overhead of ICN Storage

In this section, we provide a simple calculation to analyse, in which conditions ICN makes sense in terms of energy-efficiency. This calculation is based on the assumption to use CCN as proposed by [9]. Whereas CCN is also considered as an approach with coupled name resolution and routing/forwarding of data, other decoupled approaches such as PSIRP [10, 11] have a preceding name resolution phase prior to data forwarding. The name resolution phase might add some slight overhead in terms of hops to be traversed compared to a situation, where CCN Interest and Data messages are transferred along the shortest path between requesting client and server/cache providing the requested content object.

In the following, we assume a scenario (Fig. 2) with several clients, an intermediate CCN router, and a server. Further, the clients are all M hops away from the CCN router, which can serve the clients' content requests from its cache, and N hops (M < N) away from the server. The distance between server and CCN router is N-M. There are two options:

- No CCN support at all
- CCN support in the CCN router

For both options there are costs for the memory of the original content source. So, we can neglect those costs further, since they are the same in both options. We assume that there are K requests for the same content object by any of the clients from the server. Additional assumptions are that we have a rather optimal cache replacement strategy in the router. The additional required energy for serving K requests from the original content source is

$$E_{read\_from\_source} = K * N * E_{link} * datasize$$

The additional required energy for serving the request from the cache is

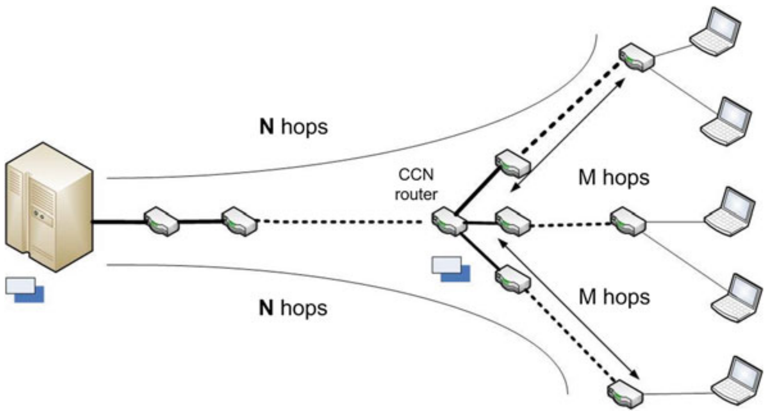


Fig. 2. Network scenario for ICN energy analysis

$$E_{read\_from\_cache} = K * M * E_{link} * datasize + T_{in-cache} * P_{storage} * datasize$$

ICN can help to decrease energy costs, if $E_{read\_from\_cache} < E_{read\_from\_source}$ This means:

$$\frac{K * (N - M) * E_{link}}{T_{in-cache}} > P_{storage}$$

$$\frac{(N - M) * E_{link}}{P_{storage}} > \frac{T_{in-cache}}{K}$$

Where:

$T_{in-cache}$: lifetime a content object is usually stored in the cache,
$P_{storage}$: power required to store a certain amount of data,
E_{link} : energy to transport a certain amount of data across a link.
$\frac{T_{in-cache}}{K}$ can be replaced by the average time between two accesses to the content object. We call this time the inter access time ($T_{inter-access}$). The access frequency (f) can be defined by $= \frac{1}{T_{inter-access}}$.

Thus, the equation above becomes to

$$\frac{(N - M)E_{link}}{P_{storage}} > 1/f$$

$$f > \frac{P_{storage}}{(N - M) * E_{link}}$$

As an example calculation we take the following numbers from above, with E_{link} = 15 Joules/Gigabit = 120 Joules/GB storage power = 3 W/GB for storage in DRAM, and assume that M = 1, N = 11. In this case, it can be shown that ICN only reduces energy consumption if f > (3 W/GB)/((11-1) * (120 Joules/GB)) = 1/400 s.

This means that each content object should be accessed at least once from the cache within 400 s. This is equivalent to *9 accesses per hour*. Otherwise, ICN might not be energy efficient. In addition, there might some queuing issues due to energy costs for serving packets in the router queues. For example, if the K requests were served and the content objects were transmitted in parallel, then the calculation above would be valid again. If there is buffer (not cache) sharing and/or link sharing, then we might need to consider energy consumption for the buffer as well (in addition to $T_{in-cache}$). This energy consumption is different for different content objects, depending on when they arrive at the buffer (after fetched from the cache and before going out to the link). But again, there is also queuing delay (and consequently energy costs) in traditional CDNs. It would be interesting to evaluate and compare the impact of queuing effects on ICN and other traditional CDNs.

2.4 Energy Efficiency and Performance Trade-off in ICN

The calculations above are a pure energy consumption perspective. However, to evaluate overall system performance, trade-offs between energy efficiency and performance could be of special interest. Note that one important aspect and also an advantage of CCN is to reduce networking delay, i.e., to serve the content as fast as possible, especially for delay-sensitive applications. In this perspective a useful and widely accepted metric to consider is the energy-delay-product. So even in the case of possibly higher energy consumption with respect to traditional CDNs, ICN might still be a useful approach for delivering data in terms of overall system performance.

3 Summary and Outlook

This paper discussed energy efficiency issues in ICN. We provide an investigation on the impact of ICN architectures on energy consumption of networking hardware devices and compare them with the energy consumption of other content dissemination methods. In particular, by some preliminary investigation, we showed the consequences of caching in ICN from the energy efficiency perspective, taking into account the energy consumption of different hardware components in ICN architectures. Based on the results of the analysis, we address the practical issues regarding the possible deployment and evolution of ICN from an energy-efficiency perspective. There are still several open issues that require further investigation:

- The paper [1] deals with issues of CCN in fixed networks using simple calculations based on edge and core routers' energy consumption and trace-based simulations. Since edge routers tend to spend more energy per transferred bit than core routers, it is argued that CCN trying to bring routers close to content sources (using appropriate caching strategies) will help to reduce the overall energy consumption of the Internet. Issues remaining for CCN in fixed networks are power-aware routing traffic, virtualization in CCN, and how to achieve/ensure energy proportionality (avoiding energy loss during idle time).

- How the above mentioned issues can be done in CCN for mobile/wireless networks can be considered as an open research issue. Furthermore, the analysis presented in [1] is rather based on brochure-like assumptions, no actual traffic modelling was mentioned. To have better understanding about the energy consumption with real/simulated network traffic included, a more detailed analysis would be required. A recent paper [12] substantiates the calculations and is a good starting point for further analysis.
- Optimal caching strategies of content objects for energy efficient content retrieval in ICN (also trade-off between delay/performance and energy consumption) have to be investigated. Cache replacement strategies should explicitly consider energy costs in addition to performance-related parameters.
- There are several transport layer issues in ICN from the energy efficiency perspective. What transport protocol should be used for energy-efficient data transfer in ICN? Can TCP still be used? If so, what phases can remain, what phases need be modified? Is coordination between TCP and edge/green routers needed? If so, how?
- The calculations presented in this paper should be extended to include queuing delay (and probably for different caching strategies) to quantify the energy costs of storage/transmission in ICN (in comparison with traditional Content Distribution Networks (CDNs)). More advanced analytical models have to be developed. Simulations for larger network scenarios with different parameters for network topologies, content access models, content popularity, etc. have to be performed.
- The various solutions for the integration of ICN into wireless networks require further evaluation using both simulation and real-world testbeds.
- Network coding might be useful for ICN in mobile wireless networks but it also requires additional caching decisions for withholding data in the memory. Consequently, feasibility analysis of network coding for mobile/wireless ICN might be a research challenge (decision on caching time in the memory with respect to performance/energy efficiency).
- As proposed in this paper, ICN might be a networking technology to support cloud computing. Integration of ICN into data centres and possibly end systems should be investigated by means of real prototype implementations. Different approaches, e.g., coupled and decoupled ICN architectures, need to be investigated further.

References

1. Lee, U., Rimac, I., Kilper, D., Hilt, V.: Toward energy-efficient content dissemination. IEEE Netw. **25**, 14–19 (2011)
2. Chamara, G., Christensen, K., Nordman, B.: Managing energy consumption costs in desktop PCs and LAN switches with proxying, split TCP connections, and scaling of link speed. Int. J. Netw. Manag. **15**(5), 1099–1190 (2005)
3. Lee, J., Kim, D.: Proxy-assisted content sharing using content centric networking (CCN) for resource-limited mobile consumer devices. IEEE Trans. Consum. Electron. **57**(2), 477–483 (2011)

4. Ko, B.J., Pappas, V., Raghavendra, R., Song, Y., Dilmaghani, R.B., Lee, K.-W., Verma, D.: An information-centric architecture for data center networks. In: ICN'12 Workshop, co-located with ACM SIGCOMM, Helsinki, Finland, 17 Aug 2012
5. Perino, D., Varvello, M.: A reality check for content centric networking. In: Proceedings of the ACM SIGCOMM Workshop on Information-Centric Networking (ICN '11), pp. 44–49. ACM, New York (2011)
6. Careglio, D., Da Costa, G., Kat, R., Pierson, J.-M.: Hardware leverages for energy reduction in large scale distributed systems. Technical report, IRIT-2011-1-FR
7. Siris, V.: Content-centric networking architectures for moving objects, short term scientific mission - Final Report, EU COST Action IC 0906, 12 July 2012
8. Wong, W., Nikander, P.: Secure naming in information-centric networks, ACM ReArch 2010, 30 November 2010, Philadelphia, USA
9. Jacobson, V., Smetters, D.K., Thornton, J.D., Plass, M.F., Briggs, N.H., Braynard, R.L.: Networking named content. In: ACM CoNEXT 2009, Rome, December (2009)
10. Trossen, D., Parisis, G.: Designing and realizing an information-centric internet. Commun. Mag. IEEE **50**(7), 60–67 (2012)
11. Xylomenos, G., Vasilakos, X., Tsilopoulos, C., Siris, V., Polyzos, G.: Caching and mobility support in a publish-subscribe internet architecture. IEEE Commun. Mag. **50**(7), 52–58 (2012)
12. Choi, N., Guan, K., Kilper, D.C., Atkinson, G.: In-network caching effect on optimal energy consumption in content-centric networking. In: ICC, pp. 2889–2894 (2012)

Cutting Down the Energy Cost of Geographically Distributed Cloud Data Centers

Huseyin Guler[1(✉)], B. Barla Cambazoglu[2], and Oznur Ozkasap[1]

[1] Koc University, Istanbul, Turkey
hguler@ku.edu.tr
[2] Yahoo! Research, Barcelona, Spain

Abstract. The energy costs constitute a significant portion of the total cost of cloud providers. The major cloud data centers are often geographically distributed, and this brings an opportunity to minimize their energy cost. In this work, we model a geographically distributed data center network that is specialized to run batch jobs. Taking into account the spatio-temporal variation in the electricity prices and the outside weather temperature, we model the problem of minimizing the energy cost as a linear programming problem. We propose various heuristic solutions for the problem. Our simulations using real-life workload traces and electricity prices demonstrate that the proposed heuristics can considerably decrease the total energy cost of geographically distributed cloud data centers, compared to a baseline technique.

1 Introduction

Cloud data centers provide massive computing power to serve a large amount of tasks generated by the Internet services as well as IT industries. While processing incoming computational tasks, cloud service providers also need to satisfy certain service level agreements (SLAs). In order to meet the requirements of their SLAs, cloud providers geographically distribute their data centers around the world. The data centers' high computing power requires significant energy for both running the computing resources and also cooling them. The cost of energy constitutes an important fraction of the total operational costs of distributed data centers [1]. Therefore, service providers constantly chase novel methods to reduce their energy cost.

Reducing the energy consumption of large-scale distributed systems has recently been a hot research topic [1]. Some studies [2,4,9] focus on server consolidation by moving virtual machines across data centers and also consider SLA penalties if the deadline of a certain job is not satisfied [2,9]. Le et al. [8] and Gao et al. [3] concentrate on the greenness aspect of data centers and allow service providers to trade off between the electricity cost and the carbon footprint. Certain studies investigate the data center cooling problem and propose solutions based on workload placement [11,17] and thermal storage [5,18]. In recent years,

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 279–286, 2013.
DOI: 10.1007/978-3-642-40517-4_23,

researchers have also investigated the impact of spatio-temporal electricity price variations on financial cost savings [6,7,10,12–16,19,20]. All these works either do not consider the SLA requirements or ignore cooling related costs. In this paper, we propose electricity-cost-aware request dispatching algorithms which also consider SLA related penalties.

2 Problem Specification

In this section, we formally state our linear optimization problem. Table 1 lists the parameters and system variables used in the formulation.

We calculate the performance coefficient (CoP) of a data center according to its outside weather temperature. We set the CoP of the hottest data center (26 °C) as 2.0 (this means 1W energy is needed to cool down the data center for each 1W of IT job), and for the coldest data center (−9 °C) as 1.2, which is similar to the CoP of the currently existing energy-efficient cloud data centers [1].

The performance coefficient of DC i with temperature T_i:

$$CoP(T_i) = 1.2 + 0.128 \times \sqrt{T_i + 9} \tag{1}$$

Table 1. System parameters

Description	Symbol
Total energy cost of servers in DC i (\$)	E_i^{IT}
Total energy cost of DC i (\$)	E_i^{total}
Total penalty paid in DC i	Pen_i
Length of a unit time slot	u
Electricity price of DC i in time slot $[t, t+u)$	$E_i(t)$
Number of DC	N
Number of different type of servers	K
Set of servers in DC i	s_i
Set of type k servers in DC i	$s_{i,k}$
Number of CPU cores that a type k server has	c_k
CPU frequency of a type k server	f_k
Power consumption of a type k server when idle (Watt/u)	P_k^{idle}
Power consumption of a type k server at peak (Watt/u)	P_k^{peak}
Power consumption of a type k server s in time slot $[t, t+u)$	$P_{k,s,t}$
Number of jobs	J
Number of CPUs job j requires	c_j
Total number of timeslots job j requires	l_j
CPU frequency job j requires	f_j
Submission time of job j	Ts_j
Deadline of job j	Td_j
Penalty of late delivery of job j	Pen_j
Percentage of time that SLA of job j is violated	$\sigma(j)$
Number of time slots required to process job j in server s	$T_{j,s}$

The decision variable indicating whether server s is busy operating in time slot $[t, t+u)$:

$$x_{s,t} = \begin{cases} 1, & \text{if server } s \text{ is busy operating} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Power consumption of a type k server [14]:

$$P_{k,s,t} = P_k^{\text{idle}} + (P_k^{\text{peak}} - P_k^{\text{idle}}) \times x_{s,t} \tag{3}$$

$$E_i^{\text{IT}} = \sum_{k=1}^{K} \sum_{s \in s_{i,k}} \sum_{t=0}^{\lfloor T/u \rfloor} P_{k,s,t} \times E_i(t) \tag{4}$$

Total cost of data center i:

$$E_i^{\text{total}} = E_i^{\text{IT}} \times CoP(T_i) \tag{5}$$

Decision variable indicating whether job j is dispatched to server s:

$$r_{j,s} = \begin{cases} 1, & \text{if job } j \text{ is assigned to server } s \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

$$T_{j,s} = \frac{c_j \times f_j \times l_j}{c_k \times f_k}, \text{where } s \text{ is a type } k \text{ server} \tag{7}$$

Percentage of time that SLA of job j is violated [4]

$$\sigma(j, s, t) = (Ts + T_{j,s} - Td_j)/T_{j,s} \tag{8}$$

$$Pen_{j,s,t} = P_{k,s,t} \times E_i(t) \times \frac{Gom(\sigma(j, s, t))}{100} \tag{9}$$

$$Pen_i = \sum_{j=1}^{J} \sum_{s \in s_i} \sum_{t=0}^{\lfloor T/u \rfloor} r_{j,s,t} \times Pen_{j,s,t} \tag{10}$$

Our objective is to minimize:

$$\sum_{i=1}^{N} E_i^{\text{total}} + Pen_i, \tag{11}$$

subject to

$$s_i = \sum_{k=1}^{K} s_{i,k} \tag{12}$$

In other words, our goal is to minimize the cost of a provider during the given time period while respecting QoS requirements (this includes financial penalties if SLA is violated).

3 Proposed Algorithms

We have a central scheduler that receives the incoming jobs and then forwards each job to one of the idle servers. Our proposed algorithms take advantage of three important factors:

– Spatial electricity price variation.
– Temporal electricity price variation.
– Reduced cooling cost in cooler places due to evaporation.

Considering these factors, we propose two types of request dispatching algorithms. In the first type of algorithms, each incoming job request is immediately scheduled to an available server. The algorithms of the second type can schedule jobs ahead of time within a time window if they "forecast" that the electricity price will be lower in the future.

3.1 Immediate Scheduling Algorithms

The jobs are immediately scheduled in FCFS order. The following two algorithms differ in the way they decide on which server to assign a job.

Cheapest data center(`CheapestDC`): The current electricity prices of all data centers are checked and a random server is selected from the cheapest data center. If all servers in the cheapest data center are busy, then a server from the second cheapest data center is selected. The procedure continues until the job is scheduled. If all servers are busy operating at that time then the job is put in the queue again to be scheduled in the next time slot. Only the spatial electricity price change is exploited in this simple greedy heuristic.

Cheapest server (`CheapestS`): Different than the `CheapestDC` algorithm, this algorithm takes advantage of the outside weather temperature associated with the data centers. The assumption is that the total cost of running a job in a server in cooler locations can be cheaper even if the server is not located in the cheapest data center. The algorithm runs the job on the server with the lowest expected total cost.

3.2 Delayed Scheduling Algorithms

`CheapestDC` and `CheapestS` aim at scheduling the jobs in the current time slot. However, it is possible to postpone the execution of a job to future time slots if we can somehow identify that the current electricity is expensive. To this end, we can use historical electricity prices to determine whether to schedule the job immediately or delay its execution to a later time slot. As in the `CheapestS` algorithm, we examine every server for all time slots within a predefined time window and select the best server and time slot combination in terms of the electricity cost. In this case, we utilize all three variations including spatial and temporal electricity price fluctuations as well as the reduced cooling cost of the servers located in colder climates.

In addition to the novel time window approach, we also implemented an internal ordering of the jobs in this algorithm and observed the effect of the ordering on the final performance. These orderings are first come first serve (FCFS), longest job first (LJF), and shortest job first (SJF). Variations of these algorithm are named as `WindowFCFS`, `WindowLJF`, and `WindowSJF`.

Note that, in this algorithm, the scheduled time slots are fixed, i.e., we do not reschedule any job even if it is assigned to a future time slot. As a future work, we will also implement an algorithm with periodic rescheduling and compare its performance with the other solutions.

4 Simulation Setup

We simulated a geographically distributed data center network to evaluate the performance of the proposed algorithms. The simulator and the algorithms are implemented in Java. We only consider delay-tolerant batch jobs and use the Grid5000 logs for incoming job requests.[1] The job requests contain job specifications including submission time, runtime, required number of CPU cores, and similar information. Our problem formulation is generalizable to heterogeneous data centers. However, for the initial results we present here, we consider only the homogeneous data center scenario. We simulated six homogeneous data centers that are located in San Diego, California; Chicago, Illinois; Santiago, Chile; Helsinki, Finland; Dublin, Ireland and Singapore, Singapore. Each data center is given 100 servers with Xeon architecture and four core CPUs running at 2.66 GHZ [4]. We used real electricity price traces for San Diego and Chicago from FERC (Federal Energy Regulatory Commission of the USA),[2] and scaled the prices for other countries according to the country-wide average prices.[3] Average temperatures values are gathered from Wunderground.[4]

We simulated the network under three different loads: `light`, `medium`, and `heavy`. The Grid5000 data spans three years of job requests; however, we run our simulations on a weekly basis. The original log corresponds to the `light` workload. Thus, we scaled down the submission times of the jobs that belong to future weeks to construct the `medium` and `heavy` workloads. Moreover, our delayed scheduling algorithm is run under different time window values, that are 6, 12, and 24 hours to determine the best fit of the time window.

At this stage of our work, we have not yet introduced the penalty concept for the jobs and the results presented in Sect. 5 do not include any penalty-related cost. As a baseline, we also implemented a random request dispatcher (`Random`) that tries to balance the workload of the data centers. We compared the performance of the proposed algorithms against this baseline.

[1] Grid5000, http://gwa.ewi.tudelft.nl/pmwiki/pmwiki.php?n=Workloads.Gwa-t-2.
[2] FERC: Electric Power Markets, http://www.ferc.gov/market-oversight/mkt-electric/overview.asp.
[3] Wikipedia-Electricity Pricing, http://en.wikipedia.org/wiki/Electricity_pricing.
[4] Wunderground, http://www.wunderground.com/history.

Table 2. Performance of immediate scheduling techniques ($I\%$ denotes the relative percent improvement w.r.t the random scheduling baseline while C_{P}, C_{C}, and C_{T} denote the processing, cooling, and total financial cost per job, respectively)

	`Random`				`CheapestDC`				`CheapestS`			
Load	C_{P}	C_{C}	C_{T}	$I\%$	C_{P}	C_{C}	C_{T}	$I\%$	C_{P}	C_{C}	C_{T}	$I\%$
`light`	18.9	14.0	32.9	–	18.3	13.4	31.8	3.4	18.4	13.3	31.6	3.8
`medium`	12.1	8.9	21.0	–	11.9	8.7	20.7	1.7	11.9	8.7	20.6	2.1
`heavy`	8.0	5.9	14.0	–	8.0	5.9	14.0	0.0	8.0	5.9	14.0	0.0

Table 3. Performance of delayed scheduling techniques (W denotes the window size)

		`WindowFCFS`				`WindowLJF`				`WindowSJF`			
Load	W	C_{P}	C_{C}	C_{T}	$I\%$	C_{P}	C_{C}	C_{T}	$I\%$	C_{P}	C_{C}	C_{T}	$I\%$
	6	18.3	13.1	31.3	4.7	18.2	13.0	31.3	4.9	18.3	13.1	31.4	4.5
`light`	12	18.3	13.0	31.2	5.0	18.2	13.0	31.2	5.2	18.3	13.0	31.3	4.7
	24	18.2	13.0	31.2	5.1	18.2	13.0	31.1	5.4	18.2	13.1	31.3	4.9
	6	11.8	8.6	20.3	3.3	11.8	8.5	20.3	3.5	11.8	8.6	20.4	3.1
`medium`	12	11.7	8.5	20.2	3.9	11.7	8.5	20.1	4.2	11.7	8.5	20.2	3.7
	24	11.7	8.4	20.1	4.5	11.6	8.4	20.0	4.7	11.7	8.4	20.1	4.3
	6	8.0	5.9	13.8	0.9	8.0	5.9	13.8	1.0	7.9	5.8	13.8	1.4
`heavy`	12	9.1	6.7	15.8	-13.1	9.8	7.2	17.1	-22.3	8.3	6.1	14.3	-2.8
	24	9.1	6.7	15.8	-13.0	9.4	6.9	16.2	-16.4	8.9	6.5	15.4	-10.2

5 Results

As explained in Sect. 3, there are three factors that we consider: spatial and temporal electricity price change and outside weather temperature. By comparing the performance of the algorithms, we can determine the contribution of these factors to the cost saving. Tables 2 and 3 summarize our simulation results and the amount of improvement achieved by each algorithm. Most of the gain is achieved by exploiting the spatial electricity price variation as seen by the improvement of the `CheapestDC` algorithm. Next, the biggest improvement is achieved by algorithms with a time window that exploit the temporal electricity price change. The least improvement, but still significant, is by taking advantage of the reduced cooling cost of the servers located in colder climates.

In the `light` workload case, we have the flexibility to postpone the execution of a job since there are few jobs. This is also true for the `medium` workload. The current state-of-the-art data center systems work under medium workloads, where the system tries to keep the total utilization of the servers under a certain percentage, i.e., mostly around 35%-40%. When the simulation is run under the `heavy` workload, the number of jobs executed by each algorithm differs. Therefore, we presented these results in terms of cost per job. The number of jobs executed by delayed scheduling algorithms drops in the `heavy` workload case because some of the early time slots are not utilized by the algorithm, which forecasts some cheaper future time slots. In the end, we were left with many non-executed jobs in the job queue. In order to overcome this drawback, we are planning to propose a workload adaptive delayed scheduling algorithm that also utilizes the current time slots when the workload is heavy.

6 Conclusion

We proposed request dispatching algorithms that exploit spatio-temporal electricity price variations and reduced cooling cost opportunities in colder climates to minimize the energy cost of geographically distributed data centers. Our simulation results show that significant electricity cost reduction can be achieved by the proposed algorithms, compared to a random scheduler that aims to achieve only load balancing. As a future work, we plan to conduct experiments including the SLA penalty concept.

Acknowledgments. This work was partially supported by the COST (European Cooperation in Science and Technology) framework, under Action IC0804: Energy efficiency in large scale distributed systems, and by TUBITAK (The Scientific and Technical Research Council of Turkey) under Grant 109M761.

References

1. Barroso, L.A., Hölzle, U.: The datacenter as a computer: an introduction to the design of warehouse-scale machines, 1st edn. Morgan and Claypool Publishers, California (2009)
2. Buchbinder, N., Jain, N., Menache, I.: Online job-migration for reducing the electricity bill in the cloud. In: Domingo-Pascual, J., Manzoni, P., Palazzo, S., Pont, A., Scoglio, C. (eds.) NETWORKING 2011, Part I. LNCS, vol. 6640, pp. 172–185. Springer, Heidelberg (2011)
3. Gao, P.X., Curtis, A.R., Wong, B., Keshav, S.: It's not easy being green. In: Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer, Communication, pp. 211–222 (2012)
4. Goiri, I., Berral, J.L., Fitó, J.O., Julií, F., Nou, R., Guitart, J., Gavaldí, R., Torres, J.: Energy-efficient and multifaceted resource management for profit-driven virtualized data centers. Future Gener. Comput. Syst. **28**(5), 718–731 (2012)
5. Guo, Y., Ding, Z., Fang, Y., Wu, D.: Cutting down electricity cost in internet data centers by using energy storage. In: Global Telecommunications Conference, pp. 1–5 (2011)
6. Kayaaslan, E., Cambazoglu, B.B., Blanco, R., Junqueira, F.P., Aykanat, C.: Energy-price-driven query processing in multi-center web search engines. In: Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 983–992 (2011)
7. Le, K., Bianchini, R., Martonosi, M., Nguyen, T.D.: Cost- and energy-aware load distribution across data centers. In: Workshop on Power Aware Computing and Systems (2009)
8. Le, K., Bilgir, O., Bianchini, R., Martonosi, M., Nguyen, T.D.: Managing the cost, energy consumption, and carbon footprint of internet services. In: Proceedings of the ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, pp. 357–358 (2010)

9. Li, J., Li, Z., Ren, K., Liu, X.: Towards optimal electric demand management for internet data centers. IEEE Trans. Smart Grid **3**(1), 183–192 (2012)
10. Mani, S., Rao, S.: Operating cost aware scheduling model for distributed servers based on global power pricing policies. In: Proceedings of the 4th Annual ACM Bangalore Conference (2011)
11. Moore, J., Chase, J., Ranganathan, P., Sharma, R.: Making scheduling "cool": temperature-aware workload placement in data centers. In: Proceedings of the Annual Conference on USENIX Annual Technical Conference, p. 5 (2005)
12. Qureshi, A., Weber, R., Balakrishnan, H., Guttag, J., Maggs, B.: Cutting the electric bill for Internet-scale systems. In: Proceedings of the ACM SIGCOMM 2009 Conference on Data, Communication, pp. 123–134 (2009)
13. Rao, L., Liu, X., Ilic, M., Liu, J.: MEC-IDC: joint load balancing and power control for distributed Internet data centers. In: Proceedings of the 1st ACM/IEEE International Conference on Cyber-Physical Systems, pp. 188–197 (2010)
14. Ren, S., He, Y., Xu, F.: Provably-efficient job scheduling for energy and fairness in geographically distributed data centers. In: Proceedings of the 32nd International Conference on Distributed, Computing Systems, pp. 22–31 (2012)
15. Sakamoto, T., Yamada, H., Horie, H., Kono, K. Energy-price-driven request dispatching for cloud data centers. In: Proceedings of the 5th International Conference on Cloud, Computing, pp. 974–976 (2012)
16. Sankaranarayanan, A.N., Sharangi, S., Fedorova, A.: Global cost diversity aware dispatch algorithm for heterogeneous data centers. In: Proceedings of the 2nd Joint WOSP/SIPEW International Conference on Performance, Engineering, pp. 289–294 (2011)
17. Tang, Q., Gupta, S.K.S., Varsamopoulos, G.: Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: a cyber-physical approach. IEEE Trans. Parallel Distrib. Syst. **19**(11), 1458–1472 (2008)
18. Wang, Y., Wang, X., Zhang, Y.: Leveraging thermal storage to cut the electricity bill for datacenter cooling. In: Proceedings of the 4th Workshop on Power-Aware Computing and Systems, pp. 8:1–8:5 (2011)
19. Yao, Y., Huang, L., Sharma, A., Golubchik, L., Neely, M.: Data centers power reduction: a two time scale approach for delay tolerant workloads. In: Proceedings of the 31st Annual IEEE International Conference on Computer Communications, pp. 1431–1439 (2012)
20. Zhang, Y., Wang, Y., Wang, X.: Electricity bill capping for cloud-scale data centers that impact the power markets. In: Proceedings of the 41st International Conference on Parallel Processing, pp. 440–449 (2012)

Part V

Standardization Issues

Standardization Bodies, Initiatives and Their Relation to Green IT Focused on the Data Centre Side

Christina Herzog(✉)

IRIT, University of Toulouse, Toulouse, France
Herzog@irit.fr

Abstract. The aim of this paper is to provide an overview of the main relevant standardization bodies and organizations (national, international, industry groups, standards bodies and regulators) as well as projects, initiatives and actual standards in the field of Green IT, and more especially in the Data Centre area. The co-operation between these different organisations and their different point of views and working areas are also investigated. A critical analysis concerning the motivations and approaches of academia and industry towards the various standardization bodies and initiatives is given.

Keywords: Standardization · Regulation · Metrics · Technical committee · Green IT

1 Introduction

Standardisation is an important part in the field of Green IT as currently the society is consuming too much energy, and common agreements are missing. There are several organisations – international and national – working on standardization in Green IT but still common outputs are not influencing strongly enough the decision makers. A common classification of energy users is missing. This classification would lead to energy savings as international laws may be voted and penalties applied for those being not energy efficient enough. For instance, new data centres will only be constructed if they have a certain classification, old data centres may have to be renovated or pay a fee based on a certified metric. Before doing such a classification (which is beyond the scope of this paper) it is needed to understand the scope of these standardisation bodies and to identify several organisations involved in standardisation activities, both formal and informal. This includes governmental, industry-related and lobbying groups.

It is indeed more important to investigate the various motivations, the reasons of the actors being involved in standardization bodies. In particular the differences in the approaches from the academia's and industries' point of view may play an important part in the cooperation and in the – existing or not existing – standards. Obviously, one of the motivations for the industry in joining a group is to promote their own technology to become a future standard, but other motivations can also be analyzed.

J.-M. Pierson et al. (Eds.): EE-LSDS 2013, LNCS 8046, pp. 289–299, 2013.
DOI: 10.1007/978-3-642-40517-4_24,

The rest of the article is organized as follows: Section 2 will define the standard bodies, Sect. 3 talks about the existing standards, Sect. 4 focuses on Data Centres, Sect. 5 puts some critical arguments on the current situation and analyzes the motivations. Section 6 concludes the paper.

2 Standards Bodies Defined

The concept of standardisation is a flexible term when applied to (Green) IT innovation. There are strictly managed standards existing (such as those managed by the International Organisation for Standardisation (ISO), the International Telecommunications Union (ITU) and the International Electrotechnical Commission (IEC) as well as a wide variety of other de facto initiatives, metrics and frameworks being also classified as (non strictly) standards. Legislation plays also a role in the development of common approaches to (Green) IT and its initiatives should also be considered as important.

There is also a wider concept of standardisation, not governed by any overarching organisation but may be defined as how interoperable a particular technology is with the pervasive technologies in a particular market.

There are a variety of classes of initiative that can be categorised as a de facto or actual standard (metrics, frameworks, projects). These various classes of standard types are also administered by a range of different organisations from formal government-backed groups, professional bodies, and principally supplier-led organisations.

Table 1 shows standard body types, their nature of standards and is giving some examples. The standard body type section shows which kind of body it is, who can join, while the second colon what these bodies are providing, how they are structured and what they can provide for the standardization and in the last colon you find some examples of bodies, organisations, initiatives and an example of an EC project, which can also be considered has having influence in the standardization of energy saving.

To help understand the links between the different stakeholders, the Fig. 1 is giving an overview.

On Fig. 1, we can see that the first providers of materials and tools that may make their way to actual standards are industry alliances, academic researchers, or both in

Table 1. Classification of standards organisations

Standard (de facto or actual) body type	**Nature of standards**	**Example**
International, regional and national and government-backed standards bodies	Highly structured standards, requiring some degree of certification/enforcement	ISO, UN ITU, IEC, CENELEC-CEN, ETSI
Institutes and professional bodies	Structured and de-facto frameworks, metrics, and projects. Some certification required	BCS, IEEE, ASHRAE, Uptime Institute
Supplier and industry groups	De-facto standards, projects and other initiatives.	The Green Grid, Open Compute Project
Technical initiatives, projects, supplier product development,	Loosely structured initiatives, published research etc.	EC projects: e.g. CoolEmall

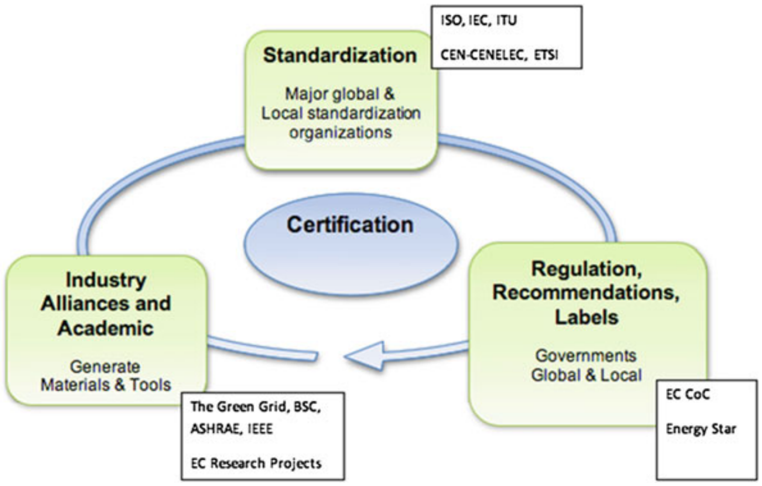

ISO, IEC, ITU
CEN-CENELEC, ETSI
Standardization
Major global & Local standardization organizations
Certification
Industry Alliances and Academic
Generate Materials & Tools
The Green Grid, BSC, ASHRAE, IEEE
EC Research Projects
Regulation, Recommendations, Labels
Governments Global & Local
EC CoC
Energy Star

Fig. 1. Existing links between standardization stakeholders

collaborative projects. Some of the proposed ideas may be presented in one or several standardization bodies to eventually become standards. These standards can in turn be used by governments (national, federal or European levels) as regulations in laws that must (and can) be enforced.

Direct usage of the materials can also be used directly by governments as regulations, recommendations or labels. While the process for formal standardization make take long since a consensus have to be achieved between all members (especially states), the direct link with governments is sometimes more efficient.

Finally, it must be noted that some materials provided by the industry and academic are used directly by final users and may become de facto standards.

In the centre of the Figure is the certification authority whose role is to certify that the measurements, claimed by suppliers of technologies follow actually the standards, the labels or the recommendations.

3 Relevant Sustainability Standards

Sustainability standards, metrics, frameworks and initiatives affect data centres in multiple ways. Some, such as green building certifications, allow data centres to voluntarily demonstrate a certain level of environmental performance. For instance BRE Global proposed a new standard for data centres building. Other standards, such as American Society of Heating Refrigeration and Air-conditioning (ASHRAE)'s temperature standards, can carry the force of de-facto mandates, or form the basis for true mandates later. Sustainability standards can affect how strongly data centres demand efficiency technologies, and sometimes even which technologies they choose. There are standards that address both IT and data centre facilities. Industrial bodies, professional groups develop and propose some standards. When a large community accepts these, they can be promoted to national and international standardization bodies, as it is at the moment discussed for the PUE (Power Usage Effectiveness) standard [6]. Government can use these defined standards to enforce regulations, as for instance for the operation of data centres with the Data Centre Code of Conduct of the European Union (voluntary based at the moment) that could form a base for a law in the future.

Several international and governmental standard bodies have been created. These groups are not always only related to IT but there is a level cooperation between different bodies and there appears to be progress in the international standardisation in the sector of IT.

ISO is made up of numerous country-specific standards bodies. Inside ISO, Technical Committee TC207 is interested in environmental management. IEC is particularly interested in the electro-technical field: Therefore their impact on Green IT cannot be ignored. Technical Committee TC111 has the responsibility of environmental standardization aspects. Like ISO, IEC is built on national standard bodies. A Joint initiative between IEC and ISO (JTC1/SC39) is investigating together Green IT.

ITU has developed standards, not only but mainly in the telecommunication sector, and is therefore also influencing the IT. ITU is built on direct national influence (members are nations, not national bodies like for ISO and IEC). In ITU, Study Group 5 is in charge of environmental aspects.

Altogether, these three bodies develop International Standards that are fully consensus-based and represent the needs of key stakeholders of every nation participating in these bodies. Every member country (or national body) has one vote and a say in what goes into an International Standard. Members come from all around the world. While each member is different, they do have one thing in common: all of them represent the entire range of interests in and for their country, companies and businesses, industry associations, educational bodies, governmental and regulatory bodies. All stakeholders are brought together through the country's member National Committee.

It should be noted that the most active members of these committees are mainly coming from the private sector, so their standards are recommendations and have to be seen in a critical way as private companies may want to push their technology in order to be number one on the market.

Generally it can be said that there is an overlap between these different groups coming from different areas. The positive aspect is that these different groups may have some overlap in defining standards, they may have the same direction of e.g.: saving energy, the difficulty might be that there are too many organizations, initiatives existing and that the exchange between is not smoothly, sometimes information are even not published and of course it is easy to lose the overview of activities in different initiatives.

4 Links Between the Standardization Bodies Related to Energy Efficiency in Data Centres

Joint Technical Committees are established between ISO and IEC in specific areas. For instance, JTC 1/SC 39 is the joint sub-committee on "Sustainability for and by Information Technology". The focus is on standardization related to the intersection of resource efficiency and Information Technology, which supports sustainable development, application, operation and management aspects, is investigated. Especially the WG 1 is working on data centre energy efficiency. They are responsible for the first drafts of the standard for metrics for assessing the energy efficiency of data centres (the PUE standard at the moment).

The European Commission has established a standardisation mandate. It requests that the three European standards bodies CEN, CENELEC and ETSI develop standards that enable efficient energy use in fixed and mobile information and communication networks.

CEN-CENELEC (European Committee for Electrotechnical Standardisation) embeds Technical Committee 215 on Electrotechnical aspects of telecommunication equipment. The standards produced by TC 215 are used by a variety of customers including planners and installers of information technology cabling and of those facilities containing significant concentrations of information technology equipment (e.g. data centres), manufacturers of cabling systems and associated components as well as test houses. CENELEC has a close cooperation with its international counterpart, the International Electrotechnical Commission (IEC). This close cooperation has resulted in some 76% of all European standards adopted by CENELEC being identical or based on IEC standards.

ETSI (European Telecommunications Standards Institute) produces globally applicable standards for Information & Communications Technologies. ETSI performs energy efficiency related work in support of European Commission Mandates.

Looking at a bird eye, concerning European standardization activities on Data Centres: the Network is done by ETSI, the Power infrastructure by CENELEC, the IT management by CEN, the cooling by ASHRAE and the monitoring by CEN/CENELEC. The need for having joint and coordinated groups is therefore obvious. The establishment of the Coordination Group on Green Data Centers (CEN-CENELEC-ETSI) helps to harmonize initiatives. Nevertheless, for researchers in the field, the challenge is big to understand and follow all individual developments.

4.1 Regulations

Through initial work such as the Directive on Energy End-Use Efficiency and Energy Services, National Energy Efficiency Action Plans (NEEAPs), the ICT for Energy Efficiency (ICT4EE) Forum, the European Commission is attempting to establish a common and worldwide methodological framework by the whole ICT sector for the measurement of its energy and carbon footprint. For instance, established by the European Commission and parties from the IT industry in 2010, the ICT4EE forum focuses on two key aspects of Eco-efficient IT: first, how the technology industry can curb its energy use; and second, how it can help other sectors do likewise. By mid 2010, four industry associations had signed up to represent the European, Japanese and American ICT industries in the Forum: DigitalEurope; Global e-Sustainability Initiative (GeSI); the Japanese Business Council Europe (JBCE); and TechAmerica Europe. Works may one day lead to worldwide regulations.

The European Code of Conduct on Data Centres Energy Efficiency has been developed in response to the increase in energy consumption in data centres and the current needs to decrease the economic, environmental and energy supply security impacts. The aim is to inform and foster the improvement of energy efficiency in the planning and operation of data centres. The Code of Conducts aims to achieve this by raising awareness and recommending energy efficient best practices and targets [3].

The Code of Conduct it is not a legally binding document but a voluntary initiative with the objective of bringing stakeholders together. Parties signing up will be expected to follow this set of best practices recommendations and abide to the principles described therein. The Code contains a comprehensive list of best practices as well as documentary aids and measurement procedures. Data centres may be entitled to use the Code logo if such improvement programs have been certified by appropriated certification authority and recognised by the EU Commission [4]. If the Code of Conduct works well, it could be made mandatory under European law to encourage energy efficiency among non-participants; conversely, if it doesn't produce results, the European Commission might seek a tougher approach.

Energy Star program started in 1992 by the US EPA (Environment Protection Agency, www.energystar.org). Since 2009 only, specifications for servers are available, and in early 2013 active state, performance reporting were added, together with blade and multi-node servers idle requirements. These steps are very important in the case of data centres since these technologies are now days at the core of their operation. While being only a label and raising a lot of controversial issues on its delivery, it has the advantage of being now worldwide known and applied.

As we can see, no real regulations exist today and no standards made its way to a formal regulation that could be enforced by law with incentives or penalties applied.

4.2 Industry Groups and Professional Bodies

Writing a report about standardisation there is also the need to take the institutes and professional bodies, groups of parties, into account and to investigate more deeply about their activities and their influence on the on-going research and on the society. These groups differ in several dimensions: Some are country based, others are at European or global levels; some are activated by governments while others are industry or professional based; some provides standards, others certifications.

The Green Grid is a non-profit, open industry consortium of IT suppliers, end-users, policy-makers, technology providers, facility architects, and utility companies. The aim is to promote the agenda of these suppliers but also unite global industry efforts, create a common set of metrics, and develop technical resources and educational tools. The Green Grid has expanded its mission from "energy efficient IT" to "resource efficient IT," meaning that it will begin looking at water, carbon, materials, waste, in addition to just energy. It is linked to the global ecosystem of technical organisations and government institutions: The American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE), The British Computer Society BCS-The Chartered Institute for IT, China Communications Standards Association (CCSA).

The Green Grid has developed a number of metrics, frameworks and initiatives in the domain of Data Center Energy Efficiency: PUE, WUE, and CUE, to name a few. These metrics assess the quality of a data centre in terms of efficiency of the resource provided against the resources useful for the IT equipment. The PUE metric is also used in the European Code of Conduct (CoC). Many data centre service providers now report that customers are asking PUE numbers in procurement documents. It should be stressed that there is no regulatory agency that monitors or certifies PUE ratings, and therefore the figures, widely cited, have no legal status, and are prone to distortion by technical and marketing staff [5].

However, we can note that two task forces, namely the Data Center Metrics Coordination Task Force (U.S. Regional Task Force) and the Global Harmonization of Data Center Efficiency Metrics Task Force (Global Task Force), have affirmed PUE as the agreed-upon metric for measuring infrastructure energy efficiency in data centres. These task forces are composed of industry players and government agencies, from US and Japan. In a published memo, the Data Center Metrics Coordination Task Force not only affirms agreement for PUE, it also provides recommended calculation and reporting guidelines for PUE.

It can be noted the PUE metric is currently being proposed for standardisation at the ISO level (a process that should be completed in approximately two years). Altogether from the first mentioned to this metric in 2007, it will take about 8 years to become a standard, which is an eternity in the timeline of data centres developments.

Started in 2007, Climate Savers Computing Initiative (CSCI) is a non-profit group of consumers, businesses, and conservation organisations dedicated to reducing the energy use and CO2 emissions from computers. In 2012, it became part of Green Grid. Computer and component manufacturers participate in Climate Savers by committing to make products that meet or exceed current ENERGY STAR standards. Consumers and corporations participate by committing to choose PCs and volume servers that meet or exceed the standards, and to use power management features. CSCI's "baseline" efficiency standards for PCs and servers are based on the respective ENERGY STAR standards for PCs and servers. Higher levels of CSCI certification (Bronze, Silver, Gold) add additional requirements regarding power supply efficiency. At the electronic level, the EPEAT (Electronic Product Environmental Assessment Tools) has the same objective, i.e. to help identify greener computers and electronic equipments.

The Uptime Institute (UI) provides education, publications, consulting, certifications, conferences and seminars, independent research, and thought leadership for the enterprise data centre industry and for data centre professionals.

The British Computer Society has a number of initiatives – particularly through its Data Centre Specialist Group (DCSG). In collaboration with the UK Carbon Trust, the BCS DCSG developed open source software that can be used to model energy efficiency and carbon emissions in data centres on a per-service basis (leading and transferred to the commercial Prognose software). The simulation tool has been developed by some of the advisors to the EC on how to measure data centre efficiency. They developed a set of metrics (DC-FVER: Data centre Fixed to Variable Energy Ratio metric). Although there has been no indication of this, the EC could recommend the use of such metrics in a future iteration of the data centre Code of Conduct.

The American Society of Heating, Refrigerating, and Air-Conditioning Engineers (ASHRAE) is an influential standards group whose work impacts data centres in several ways. ASHRAE's guidelines have a significant influence on data centre design and equipment selection, and any changes are inevitably controversial. One set of ASHRAE guidelines describes the temperature and humidity ranges in which data centre computer rooms should operate. Data centres are intended to operate most of the time within ASHRAE's fairly narrow "recommended" temperature and humidity ranges. For limited periods of time, ASHRAE permits IT equipment to venture outside this recommended range into a broader "allowable" range of temperature and humidity conditions. They create new, broader allowable ranges for data centres that wish to aggressively prioritize energy efficiency (at the cost of potentially higher operational risk). The new ASHRAE allowable ranges give additional latitude to data centre operators that wish to maximize use of economizers and "free cooling". Rather than using energy-hungry mechanical cooling systems, economizers allow data centre to use cool outdoor temperatures to meet some or all of the data centre's cooling demand. This flexibility allows data centres to reduce energy use or even design facilities with no mechanical cooling at all ("chiller-less" designs). Although ASHRAE standards do not have the force of law, many jurisdictions adopt them as part of mandatory building codes. Thus, ASHRAE's 90.1 standards strongly affect new data centre construction and major renovations.

The Open Compute Project (OCP) is an initiative from Facebook to build one of the most efficient computing infrastructures at the lowest possible cost. Custom designing, building own software, servers and data centres from the ground up, and finally share those technologies as they evolve is the objective of this project.

Of particular interest are two main aspects:

- Hardware management: Scale computing requires a small and stable set of tools to remotely management machines. The hardware management specification incorporates a set of existing tools and best practices and leverages existing tools for remote machine management.
- The Open Rack is the first rack standard that is designed for data centres, integrating the rack into the data centre infrastructure, part of the Open Compute Project's "grid to gates" philosophy, a holistic design process that considers the interdependence of everything from the power grid to the gates in the chips on each motherboard.

It can be noted that the specifications are free of charge to download, as part of this open source initiative.

5 Critical Points

What can be clearly seen is that standardization in IT is a complex system of different components, various organizations, bodies and different points of view but it is quite obvious that there is a growing interest in standardization by universities, industries and governments. Unfortunately the reasons for it are very different and this is also the reason why so many different bodies, organizations and initiatives exist [1].

Comparing the different aims, approaches, duties, priorities, … between industries and universities it is obvious that even if they join the same standardization bodies their interests can't be the same [2].

Table 2 below shows the different interests of academia and industry divided into 3 sections:

- Research and innovation
- Criteria of dissemination
- Organization

Table 2. Similarities and differences of academia and industry

			Academia	Industry
Research and Innovation	A	Duties/ responsibilities	Common welfare Extension of knowledge	Profit orientation Offering service and prodducts for the market
	B	Core competence	Fundamental research in Software and Hardware User oriented research Experimental research	User oriented research Experimental research Product development
	C	Approach	Search and find General	Decide and act concretely
	D	Priorities of topics	Personal interests Expected appreciation Financing	Strategic development of the company Portfolio of the products
	E	Selection of topics	Autonomic funding relevant	Innovation management Top management
			Academia	**Industry**
Criteria and dissemination	F	Criteria of efficien-cy	Scientific reputation	Profit and company value
	G	Criteria of quality of the work	Systematic production Reconstructable processes and re-sults Big application area Explanatory Contribution	Usability of the results Big effects for the clients usage Advantageous eco-nomic solution for a concrete application area Production of an inno-vation leading to a temporary monopoly position
	H	Reference groups	Scientific community Student	Clients Other units within the company
	I	Distribution of the results	Conferences Publications Patents	Products Internal processes Services Patents

Organization	J	Freedom of action	High Limited through resources (funding, staff, equipment, …)	Average limits through management
	K	Funding	Non-performance related basic financing Calls of funding organizations Services for companies	Budget of the innovation management In house accounting
	L	Organizational framework	Fixed and solid Influenced through scientific community Need safety concerning the expenses	Flexible Influenced through market needs, clients' needs Searching for information about efficiency and risk
	M	Relation with other units of the organization	Limited administrative support is offered Interaction within a given framework Parallel units with other fields of competences	Part of a chain within the company Targets given by the management

In each of these 3 parts there are different sections and the different approaches, the different interests of academia and industry are shown. It is obvious that in a standardization body where the number of academia and industry partners is equivalent the outcome is more balanced in terms of innovation and interest of the industry. While when the number of industries in a body is leading the recommendations for standards, the outcome might be more business related and less influenced by the latest research.

In the line C concerning the approach the operating research and development departments in companies prefer simple solutions. Solutions, which are not costly, working immediately and non high-maintenance products are definitely the ideal outcome, the perfect serviceable knowledge. Even if with more investments e.g. measuring the load of servers, or thinking about different methods of cooling the companies could achieve better solutions they keep in mind the profit and the management is setting the priorities – see line E.

This approach is also reflected in the decisions of the standardization bodies and also in the decision which body should the industry join. Often it is the easiest way to do lobbying for the solution, which is provided already by the company, but it might not be the best one for the sustainability.

Line K shows one of the difficulties for universities as they have a limited budget and often no funding for joining standardization bodies as getting access to documents and meetings is sometimes limited to paying partners. This limits the accessibilities for research institutes and as a result the outcome, the recommendations are not including the latest research.

Due to the differences of industry and academia standardization bodies and organizations are limited in their actions and in producing results, recommendations, which may influence governments or give support for further actions like launching open calls.

6 Conclusion

Having a look on different standardization bodies and organizations it is necessary to investigate more deeply about their structure, their members and their aims. Currently some of the organizations are not accessible easily, and for evaluating their recommendations it is necessary to become a paying or active member. This makes it even more difficult to use their results for e.g. evaluating a data centre or following their recommendations for research. The question which has to be faced is: How can recommendations be followed if there is no background information available? There are so many standardization bodies and influence groups on the market with different aims, often unknown, with different parties involved, but without exchange. Every organisation, every government following the recommendation of one of these bodies has to be aware that there is no outside control, and that these bodies are not necessarily objective organisations.

Acknowledgment. This research work is partly supported by the European Commission under contract 288701 through the project CoolEmAll and by the COST (European Cooperation in Science and Technology) framework, under Action IC0804.

References

1. Blind, K., Gauch, S.: Trends in ICT standards: the relationship between European standardisation bodies and standards consortia. Telecommun. Policy **32**(7), 503–513 (2008), ISSN 0308-5961. Elsevier
2. Jakobs, K., Procter, R., Williams, R.: The making of standards: looking inside the work groups. IEEE Commun. Mag. **39**(4), 102–107 (2001)
3. Uddin, M., Rahman, A. A.: Energy Efficiency and low carbon enabler green IT framework for data centers considering green metrics. Renew. Sustain. Energy Rev. **16**(6), 4078–4094 (2012)
4. European Commission.: The EU-Code of Conduct on Data Centers Energy Efficiency (2008)
5. Federal Ministry for the Environment, Nature Conservation and Nuclear Safety. Energy-Efficient Data Centers: Best-Practice Examples from Europe, The USA and Asia (2010)
6. Belady, C. (ed.): The Green Grid Data Center Power Efficiency Metrics: PUE and DCIE, The Green Grid, White Paper #6 (2008)

Towards Service Orchestration Between Smart Grids and Telecom Networks

Sergio Ricciardi,[1(✉)], Germán Santos-Boada[1], Miroslaw Klinkowski[2], Davide Careglio[1], and Francesco Palmieri[3]

[1]Technical University of Catalonia – BarcelonaTech (UPC), Barcelona, Spain
{sergior, careglio, german}@ac.upc.edu
[2]National Institute of Telecommunication (NIT), Warsaw, Poland
m.klinkowski@itl.waw.pl
[3]Second University of Naples (SUN), Aversa, Italy
fpalmier@unina.it

Abstract. In the last years, the research efforts in smart grids (SG) and telecommunication networks (TN) have been considerable but never converged to a common view and, due to the lack of strong interactions between the two worlds, only limited benefits have been achieved. We envision in this paper that future TN (as well as any other ICT application) will interact with the SG, enabling (1) the TN to know the energy source and cost that is currently powering its equipment, (2) to turn the TN into an active client which can request to the SG the quantity and quality (e.g. green) of energy that it needs, and (3) a service orchestration between SG supply system and TN operations. As a consequence, the enabled interoperability between TN and SG would allow TN to take energy-aware management decisions in function of energy-related information provided by the SG. For example, TN can route packets with the objective of optimizing green criteria, while SG can route the energy towards the TN clients with the objective of not wasting surpluses of green energy. These new energy and data routing capabilities can be exploited not only by SG operators and telecom carriers but also by any energy consumer/producer within the ICT world. This may include industry and institutional ICT premises, datacenters, home automation, wireless and mobile cellular networks, which will be able to implement their own energy-aware management and operations (M&O) by considering the quantity, quality and cost of the energy currently provided by the smart grid.

Keywords: Telecommunications networks · Smart grids · Service orchestration · Energy efficiency

1 Introduction

In the last years, larger and larger demands in terms of both network connectivity and energy provisioning have being fostered by the astonishing development of the Information and Communication Technologies (ICT). The ever-increasing data volumes to be processed, stored and accessed every day within the modern Internet-based infrastructures, empowered by ultra-high speed communication networks, result in the ICT energy demand to grow at faster and faster rates. For this reason, energy-oriented networking practices are being investigated in order to lower the ecological footprint of modern communication infrastructures.

However, since the electrical energy needed to power ICT devices is not directly present in nature, it has to be derived from primary energy sources, i.e. from sources directly available in nature, such as oil, sun, nuclear, etc. Some of them are renewable, since they come from natural *flows* (like sunlight, wind and tide), regenerating on a relatively small time scale, while others are not-renewable, since they come from specific natural *storages* (like fossil fuels or nuclear), which take eras to form [1]. The scarcity of the traditional fossil energy sources with the consequent rising energy costs have become one of the major challenges for the Information and Communications Society (ICS). Therefore, as part of the anthropological ecological footprint, the energy consumption and the indirect GreenHouse Gases (GHG) emissions are now considered as new constraints for ICT. Nevertheless, the ICT sector has the ability to reduce its ecological footprint (and hence its energy consumption and GHG emissions) through the use of innovative technological solutions [2].

For these reasons, new energy management and distribution paradigms are emerging, based on the concept of Smart Grid (SG), introducing full control, as well as adaptability and dynamism on the exploitation of energy sources, in order to make the most from the available options and drive the change toward a more sustainable society.

The purpose of this work in progress article is to analyse the possibilities offered by the SGs and, in particular, Microgrids and their interoperability with the Telecommunication Networks (TNs). This work aims at illustrating (1) the energy-follows-data and (2) the data-follow-the-energy techniques, and (3) setting the bases for future research that may unify the energy-follows-the-data and data-follow-the-energy into a common *energy-oriented SG&TN paradigm* with the potential to become a new reference architecture for SGs deployments supporting the ICT world.

2 Technological Background

2.1 Smart Grids

A SG [3] is an electrical grid that uses ICT to gather and act on information related to the generation, transmission, distribution and consumption of energy in an automated fashion in order to improve the efficiency, reliability, economics and sustainability of the whole energy process. Classic grids were designed for one-way flows of electricity, whereas a SG is able to handle in a better way bi-directional energy and information flows between the consumer (industrial and/or private users) and the grid, allowing for distributed power generation from photovoltaic panels on building roofs, fuel cells, charging to/from the batteries of electric cars, wind turbines, pumped hydroelectric plants, and other sources. It is therefore emerging as promising solution both to achieve drastic reductions in GHG emissions and to cope with the growing power requirements.

SGs promise to change the traditional energy production/consumption paradigm in which one large energy plant provides the whole region with energy, towards a configuration in which many (small, renewable and differentiated) energy plants interchange the energy with the power distribution grid. Such *microgrids* produce their own energy and release the excesses of (green) energy to the SG, which redistributes it together with the energy produced by the legacy power plants to the sites where the energy is needed or the renewable energy is currently not available.

Such a solution facilitates the integration of increasing percentages of unmanaged energy as wind and solar and the support of the energy storage capacity, supporting massive connection of electric or hybrid vehicles, both for charging and for dumping energy into the grid, as well as other potential energy accumulators such as pumped hydroelectric plants.

SGs open a new scenario in which the energy production and consumption can be closely matched avoiding peak power productions, and in which the energy quantity, quality and cost vary in function of the power plant producing it.

2.2 Microgrids

A Virtual Power Plant (VPP) [4] is a wide area cluster of distributed energy generators that are collectively run as a unique entity and controlled by a central system. A VPP enables to control several sources of energy as a single virtual entity, exploiting their own peculiar characteristic in order to deliver peak loads or load-aware power generation at short notice. In this architecture, however, the energy streams produced by the different sites are mixed all together and it is not possible to know if the energy being distributed is green or not; only the percentage of the green energy is known.

The evolution of the current electricity grids into SGs inevitably involves the introduction of new intelligent devices with local decision-making and communication capabilities. As a result, it is necessary to introduce a new generation of Intelligent Electronic Devices (IEDs) at all the grid levels with different roles within the SG, such as smart metering, protection, power switching, Remote Terminal Unit functions (RTUs), Phasor Measurement Units (PMUs), etc. The main characteristic of a SG is that it allows the distribution of electricity from suppliers to consumers and vice versa by using digital technology in order to save energy, reduce costs and increase reliability. To achieve this goal an optimum distribution of energy is required, involving the need of energy storage capabilities (something really complex and expensive) when there is a surplus or restructuring the current system in order to flexibly accommodate the demand by exploiting new SG technologies. In that sense, the deployment of small and distributed generation islands, called microgrids [5], could facilitate the development of more flexible SGs. A microgrid is a small cluster of loads and generators acting as a single system to provide electrical or thermal energy (Fig. 1). Under normal circumstances, the excess/lack of electric power will be exported/imported into/from the macrogrid (i.e. the traditional larger, centralized grid). Microgrids can be disconnected from the macrogrid [6], communicate with each other and interchange energy among them when needed. The creation of microgrids

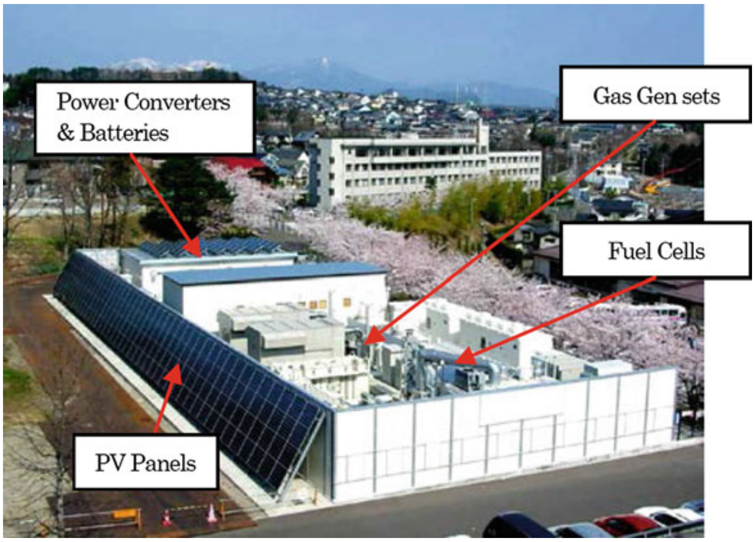


Fig. 1. Picture of the Sendai Microgrid, located on the campus of Tohoku Fukushi University in Sendai City, Tohoku district, Japan [6].

will limit the current dependency from the central power plant, increasing the reliability of the grid.

The evolution of SGs towards a fully flexible, efficient and reliable system able to interoperate with TN is expected to come through the interconnection of microgrids.

2.3 Telecom Networks

The modern Internet is connected through ultra-high speed networking infrastructures, which gobble up huge and ever increasing amounts of electricity. Furthermore, with the higher and higher demand for bandwidth, connection quality and end-to-end interactivity, network infrastructures are requiring more and more sophisticated and power-hungry devices, such as signal regenerators, amplifiers, switches, and routers. These components tend to increase the energy needs of global communication facilities. Hence, it can be easily foreseen that in the next years the Internet will be no longer constrained by its transport capacity, but rather by its energy consumption costs and environmental effects. Network equipment is foreseen to play a fundamental role in reducing GHGs emissions as it allows premises, data centres, storage and computational power to be interconnected and possibly dislocated near renewable energy plants and accessed virtually from any part of the world through high-speed connectivity. At the state of the art, miniaturization and ICT growing dynamics (i.e. Moore's and Gilder's laws) have not had the expected counterpart in power consumption reduction in the networking scenario. Miniaturization has reduced unit-power consumption but has allowed more logic ports to be put into the same space, thus increasing performances and, concomitantly, power utilization. Thus, despite architectural and semiconductor technology improvements, power consumption of network devices is still growing almost linearly with bit-rate and traffic volume. As a consequence, the total power required per a network device is exploding. It is hence necessary to adopt a systemic approach that comprises both state-of-the-art technologies improving energy-efficiency and new strategies allowing for energy-aware operation and management, acting in a cooperative fashion to achieve energy-oriented ICT [7].

3 Envisioned Scenarios

In the latest years, the efforts in the areas of SGs and energy-oriented TNs have been considerable but usually separate. At the state-of-the-art, the SG and its clients do not communicate each other (except for remote metering of consumption), and their interaction is just limited to the "blind" provisioning of raw energy from the grid to the equipment. The traditional electricity grid acts as a passive supplier, transmitting (long reach and high voltages) and distributing (short reach, middle and low voltages) the electric energy without any knowledge on the actual current energy demand of the clients. On the other side, TNs act just as passive clients of the SGs, receiving the energy without any knowledge about where it comes from, what is its environmental impact on the biosphere and actual final cost. Anyway, such information is essential to manage and operate ICT and SGs in order to exploit the renewable energy sources and lower their overall ecological footprint [8, 9]. In current SGs, the information is mainly related to the transmission and distribution infrastructure and little or no information on the electricity usage comes from the customers. Furthermore, the information provided by the grid is not used at all by the customers, who could take great advantage from the awareness of the current quantity, quality, and cost of the energy.

The enabling solution for an energy-oriented SG&TN system is to create a *bi-directional communications Interface between the SG infrastructure and the TN Control Planes* (ICP) in order to allow their interaction and the consequent optimization of the ecological footprint. The ICP will not only permit the users to adapt their behaviour, in terms of energy consumption according to the information received from the grid, but will allow them to request specific energy demands based on their scheduled activities, thus enabling bilateral dynamic interactions between the clients and energy providers. Two bidirectional flows are in fact present: the *energy flow* between the SG and the clients' premises, and the *information flow* between the SG and the clients' premises and among the different components of the SG infrastructure itself (from the energy plants to the transmission and distribution network). Smart meters at the clients' houses and PMUs in SG network provide such information which is then processed in SG control centres to manage and route the energy production and distribution in a similar way network and link state information is processed by the TNs control plane to route connection requests.

Basically, SGs will actually transform the traditional centralised electric grid in a distributed system, in which any agent that is connected to the network may provide-and-consume energy and information flows, becoming at the same time a producer and a consumer (*prosumer*). The ICP-enabled SG&TN model will add the ability to consumer to become *proactive*, allowing them to request specific energy demands to the SG. Such a distributed network will dramatically decrease the losses resulting from the long distance energy transport (data will be sent farther, not energy), and enable the easy interaction between proactive prosumers and the grid.

In such an environment, the control planes in both TN and SG manage the allocation of switching and transmission resources, both in terms of data and energy, through a proper signalling protocol. The idea is that such a communication model can be utilised by the client to: (1) know the energy source that is currently powering its equipment (e.g. green or dirty, renewable or fossil, battery or online) and (2) request a quantity and quality of energy (e.g. 10 kW of power coming from green energy source for a low priority task), so that to implement real, up-to-date, and active/pro-active energy-awareness. The key to exploit different variable energy sources is the adaptation of demand management concept where demands can be supplied *on the fly*. The ICP interface enables such dynamic adaption and unveils totally new potentials for the energy management problem, which were not possible before. In the following, we identify three scenarios that eventually, by assuming the possibility of such an interface, provide an *energy-oriented SG&TN system*.

3.1 Energy-Follows-the-Data

In the energy-follows-the-data approach, a microgrid A that has a surplus of green energy (meaning that the energy demand in its influencing area is lower than its current energy production), can be connected to an adjacent microgrid B which has a higher energy demand, so that the green energy surplus can flow from microgrid A to B and no energy has to be drawn from the macrogrid, or the current dirty energy production of microgrid B can be decreased, thus lowering its carbon footprint (Fig. 2).

This can be accomplished if the SG architecture is based on an interconnection of microgrids using energy routers [10]. These energy routers allow connecting different microgrids following circuit-switching techniques [11]. However, when changing the route of a data communication path, data are not lost, but changing the route of

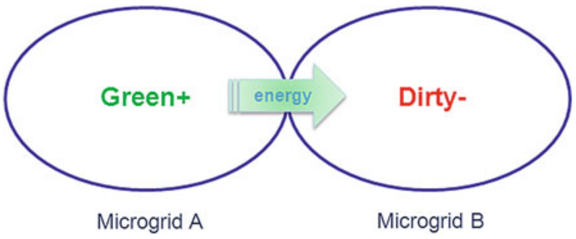


Fig. 2. The energy-follows-the-data approach: the green energy surplus of microgrid A can satisfy the power demand of microgrid B, not increasing or even lowering its (dirty) energy production.

energy, part of the energy can be lost on the way due to electric impedance and dissipation; therefore, in the energy-follows-the-data, adjacency concept and proximity of microgrids should be taken into consideration. Good results in this scenario are quite straightforward, but accurate simulation is needed to exactly quantify the achievable savings in terms of energy (resources), GHGs and money.

The energy-follows-the-data is a technique that entirely resides in the SG control plane, and no interaction is still needed with the TN control plane. In this sense, the more generic term of e*nergy-follows-the-demand* can be used.

3.2 Data-Follow-the-Energy

In the data-follow-the-energy approach (such as follow-the-sun, follow-the-wind, follow-the-tide, etc. [12]), TNs can send their data, among multiple functional-equivalent sites, to the ones that are currently powered by green energy sources. This implies a certain degree of energy-awareness of the TN control plane, since it needs to know which site is currently powered by green energy. This can be done in several ways: manually, agnostically (e.g., “blindly” following the sun according to statistical or forecasting knowledge) or automatically (e.g., employing appropriate OSPF-TE extensions to carry up-to-date energy-related information [13]).

In the data-follow-the-energy approach, the preferred sites to which data are retrieved/stored/transmitted are the ones currently powered by green (or renewable, or less costly – depending on the energy optimisation objective) energy sources (e.g. solar panels during the daytime). In this approach, the facilities, which are already powered by green energy sources, will be utilised, and no action is required in the SG control plane; this technique resides in the TN control plane.

In the example shown in Fig. 3a, the energy-aware TN control plane would choose the path passing through microgrid A to route the connection among the extreme routers; similarly in Fig. 3b, among two datacentres belonging to the same content distribution network (CDN), the one powered by green energy would be preferred.

Many techniques employing the data-follow-the-energy technique have already been employed, demonstrating the effectiveness of such an approach [8, 9, 14–16].

As shown in the example, since not only the TNs, but also other smart ICT premised can operate in such a way to selectively increase their energy demand, the more generic term of *demand-follows-the-energy* can be used.

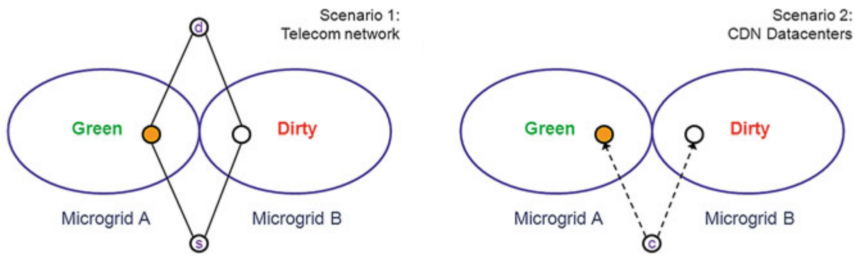


Fig. 3. The data-follow-the-energy approach in a) telecom network and b) CDN datacenters scenarios.

3.3 Service Orchestration in SG&TN System

The SG&TN service orchestration, which we propose here for the first time, consists in combining the data-follows-the-energy and energy-follows-the-data approaches. Such a system results in a multi-level network in which the distribution of energy (the *offer*) can be jointly optimised with its utilization (the *demand*) in ICT elements. Among different functionally equivalent possibilities, the TN can select a network element in a microgrid (data routing in the TN) which the SG can simultaneously power with green energy (energy routing in SG between microgrids). Figure 4 illustrates this case.

Initially, none of the two microgrid in which the TN has a router is powered by green energy. However, microgrid B is adjacent to microgrid A, which is supposed to have a green energy surplus. In such a case, the TN and the SG can act in concert through the ICP interface: the TN control plane will ask to the SG control plane green energy provisioning for its router in microgrid B, thus selecting microgrid B instead of microgrid C according to a data-follow-the-energy approach, and simultaneously, the SG control plane will provision the green energy from microgrid A to microgrid B, according to an energy-follows-the-data approach.

Note that such an optimization is possible only if the SG&TN systems operate in an orchestrated fashion.

In general, an ICT client can request an energy provisioning to the SG, specifying both the quantity and quality of the requested energy. The SG will thus fulfil or reject the request according to profitability/availability criteria. If the SG decides to accept the request, an appropriate energy path has to be established between one of the energy sources available and the site where the energy is needed. Depending on the outcome of the request, the ICT can then select the microgrid or not.

Note that, in general, the data represent the *demand* and the energy represents the *offer*, therefore configuring a smart SG&TN demand/offer matching scenario.

To make the approach viable, the following elements have to be provided:

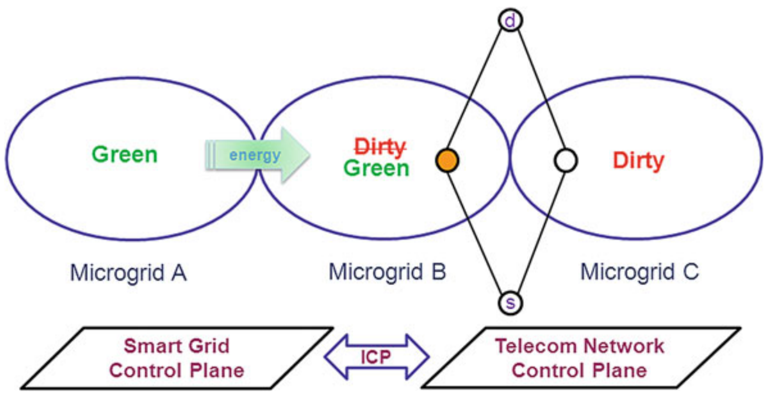


Fig. 4. Service Orchestration between SG&TN; the TN control plane selects the router in microgrid B which the SG control plane simultaneously powers with green energy of microgrid A.

(a) The ability to control the transfer of green energy between microgrids;
(b) Since the energy dissipation losses are high, the distance between microgrids must be considered as a constraint in the decision-making process;
(c) The existence of an interface between TNs and SGs and the relative policy agreements between all involved TN and SG operators;
(d) Security mechanisms and privacy issues should be addressed.

4 Conclusions

In this paper, we envision that the main progress beyond the state of the art will be the unification of the smart grid (SG) and telecom network (TN) infrastructures to exploit the potentials of the orchestrated approach. In this scenario, SG will provide the TNs with the information about the current use of the energy sources and TN will act as a proactive client of the SG, which will provide the energy when and where it is needed, and with the required output power. The interoperability between SG and TN control planes will optimize the SG&TN performance and minimize the energy requirements, GHG emissions, and energy costs. Innovative energy-aware algorithms and protocols can make possible the interaction between SG and TN via the ICP interface in a holistic systemic approach enabling previously unachievable reductions in the energy consumption of network and cloud infrastructures, towards sustainable society growth and prosperity.

Acknowledgements. This work was supported in part by the COST Action IC0804, the Spanish Ministry of Science and Innovation under the DOMINO project (TEC2010-18522) and the Catalan Government under the contract SGR 1140.

References

1. Koroneos, C.J., Koroneos, Y.: Renewable energy systems: the environmental impact approach. Int. J. Global Energy. **27**(4), 425–441 (2007)
2. SMART 2020: Enabling the low carbon economy in the information age. The climate group (2008)
3. Fang, X., Misra, S., Yang, D.: Smart grid – the new and improved power grid: a survey. IEEE Commun. Surv. Tutor. **14**(4), 944–980 (2012)
4. Lombardi, P., Powalko, M., Rudion, K.: Optimal operation of a virtual power plant. In: IEEE Power & Energy Society General Meeting, Magdeburg, Germany, July 2009
5. Colson C.M., Nehrir, M.H.: A review of challenges to real-time power management of microgrids. In: IEEE Power & Energy Society General Meeting, Magdeburg, Germany, July 2009
6. Hirose, K., Reilly, J.T., Irie, H.: The sendai microgrid operational experience in the aftermath of the tohoku earthquake: a case study. In: New Energy and Industrial Technology Development Organization (2013)

7. Ricciardi, S., Careglio, D., Santos-Boada, G., Solé-Pareta, J., Fiore, U., Palmieri, F.: Towards an energy-aware internet: modeling a cross-layer optimization approach. Telecommun. Syst., 1–22 (2011). doi: 0.1007/s11235-011-9645-7. ISSN: 1018-4864 (Springer)
8. Ricciardi, S., Palmieri, F., Fiore, U., Careglio, D., Santos-Boada, G., Solé-Pareta, J.: Towards energy-oriented telecommunication networks. Handbook on Green Information and Communication Systems, chapter 19, pp. 491–512. Academic Press, Elsevier, Amsterdam (2012). ISBN 9780124158443
9. Ricciardi, S., Palmieri, F., Torres-Viñals, J., di Martino, B., Santos-Boada, G., Solé-Pareta, J.: Green datacenter infrastructures in the cloud computing era. Handbook on Green Information and Communication Systems, chapter 10, pp. 267–293. Academic Press, Elsevier, Amsterdam (2012). ISBN 9780124158443
10. Zhu, T. et al.: A secure energy routing mechanism for sharing renewable energy in smart microgrid. In: 2011 IEEE International Conference on Smart Grid Communications, Amherst, MA, USA, Oct 2011
11. Erol-Kantarci, M., Kantarci, B., Mouftah, H.T.: Reliable overlay topology design for the smart microgrid network. IEEE Netw. **25**(5), 38–43 (2011)
12. St Arnaud, B.: ICT and global warming: opportunities for innovation and economic growth [online]. http://docs.google.com/Doc?id=dgbgjrct_2767dxpbdvcf
13. Wang, J., Ricciardi, S., Fagertun, A.M., Ruepp, S., Careglio, D., Dittmann, L.: OSPF-TE extensions for green routing in optical networks. In: 2012 Opto-Electronics and Communications Conference (OECC), pp. 411–412, 2–6 July 2012
14. Ricciardi, S., Palmieri, F., Fiore, U., Careglio, D., Santos-Boada, G., Solé-Pareta, J.: An energy-aware dynamic RWA framework for next-generation wavelength-routed networks. Comput. Netw. **56**(10), 2420–2442 (2012). doi: 10.1016/j.comnet.2012.03.016. ISSN 1389-1286 (Elsevier)
15. Wang, J., Ricciardi, S., Fagertun, A.M., Ruepp, S., Careglio, D., Dittmann, L.: Energy-aware routing optimization in dynamic GMPLS controlled optical networks. In: 14th International Conference on Transparent Optical Networks (ICTON 2012), pp. 1–4, 2–5 July 2012
16. Wang, J., Ruepp, S., Manolova, A.V., Dittmann, L., Ricciardi, S., Careglio, D.: Green-aware routing in GMPLS networks. In: International Conference on Computing, Networking and Communications (ICNC 2012), pp. 227–231, 30 Jan–2 Feb 2012

Author Index