CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2013; **00**:1–22
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/cpe

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,* [2]*Computer Science and Communications Research Unit, University of Luxembourg, Luxembourg*

SUMMARY

Virtualization is emerging as the prominent approach to mutualise the energy consumed by a single server running multiple Virtual Machines (VMs) instances. The efficient utilization of virtualized servers and/or computing resources requires understanding of the overheads in energy consumption and the throughput, especially on high-demanding High Performance Computing (HPC) platforms. In this paper, a novel holistic model for the power of virtualized computing nodes is proposed. Moreover, we create and validate instances of the proposed model using concrete measures taken during a benchmarking process that reflects an HPC usage, i.e. HPCC, IOZone and Bonnie++, conducted using two different hardware configurations on Grid5000 platform, based on Intel and AMD processors, and three widespread virtualization frameworks, namely Xen, KVM, and VMware ESXi.
The proposed holistic model of machine power takes into account the impact of utilisation metrics of the machine's components, as well as the employed application, virtualization, and hardware. The model is further derived using tools such as multiple linear regressions or neural networks that prove its elasticity, applicability and accuracy. The purpose of the model is to enable the estimation of energy consumption of virtualized platforms, aiming to make possible the optimization, scheduling or accounting in such systems, or their simulation.

Received . . .

KEY WORDS: Server Modeling; Energy-efficiency; High-Performance Computing;

1. INTRODUCTION

With the advent of Cloud Computing (CC), more and more workloads are being moved to virtual environments. Yet the question of whether CC is suitable for High Performance Computing (HPC) workload remains 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 middleware. In particular, in this study we analyse 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

*Correspondence to: University of Luxembourg, Luxembourg, 6, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg, Luxembourg. Emails: {Firstname.Name@uni.lu}

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 optimise the system by assisting scheduling or hypervisor choice, to simulate systems' future behaviour or even to account the consumed energy in case of insufficient physical monitoring infrastructure. The hardware, configuration and type of processing have an 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 [1, 2] are extended by the Configuration layer that represents the chosen middleware. Then, we propose a power model that combines multiple factors, either directly measured utilisation metrics or classifiers such as used node, hypervisor or application, consequently calling the model *holistic*. The power modelling can be lightweight in terms of creation and usage, as it is based on multiple linear regression. Similarly, modelling by neural networks is also tested, because it offers similar operational cost despite the training time that is significantly longer than the regression process. The purpose of power modelling is to validate the theoretical assumption that increasing the amount of information about the node enables better power estimation.

Compared to the related studies, our contribution in this paper can be summarised in the following elements: (1) a novel holistic model of resource allocation of virtualized computing nodes is proposed, together with exact methodology for the power consumption; (2) the model is derived using the runs of hypervisors in a concrete HPC environment, using three widespread virtualization frameworks (Xen, KVM and VMware ESXi), and two leading concurrent hardware architecture (Intel and AMD – 91% of the represented processor technologies in the Top500 November 2013 list). Our performance evaluation involves industrial reference benchmarks and does not ignore a measure of the impact on I/O operations, too often disregarded in the literature; (3) it is one of the few independent studies 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, using two independent modelling tools: multiple linear regressions and neural networks. The data used during modelling is gathered from a production environment using standard monitoring tools, proving that the model is robust and does not require high frequency and/or precision of inputs, being applicable in a wide range of systems. The obtained models present high accuracy (0.8-2.6% of cumulative error for the best models);

The article is organised as follows: Section 2 presents the background of this study and reviews related work. Our novel holistic model is detailed in Section 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, Section 5 details the experimental results obtained and the derivation of model instances by multiple linear regressions and neural networks. Finally, Section 6 concludes the paper and provides the future directions.

DOI: 10.1002/cpe

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

Table I. Overview of the considered hypervisors characteristics.

2. CONTEXT & MOTIVATIONS

In essence, Cloud middleware exploits virtualization frameworks that authorise the management and deployment of Virtual Machines (VMs). Whereas our general goal is to model Cloud systems in an HPC context, we present here the first step toward this global modelling 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 approaches of this type available today, the virtualization technology of choice for most open platforms over the past 7 years has been the Xen hypervisor [3]. More recently, the Kernel-based Virtual Machine (KVM) [4] 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 I provides a short comparison chart between Xen, KVM and VMWare 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, the experiments presented in this paper were carried out on the Grid'5000 platform [6]. 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 Section 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[†] platform [7], 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 HPC Challenge (HPCC) [8], 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

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

DOI: 10.1002/cpe

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. A view of the design choices – as regards data usage patterns – made in the development of HPCC is given in Table II.

Table II. HPCC tests locality metrics

Locality		**Spatial**	
		Low	High
Temporal	Low	RandomAccess	PTRANS, STREAM
	High	FFT	DGEMM, HPL

Bonnie++ [9], a file system benchmarking suite that is aimed at performing a number of simple tests of hard drive and file system performance.

IOZone [10], 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

The computing system power models could be classified to three groups, dependent on the level of granularity: component-level, machine-level, or distributed-system level [11]. The focus of our study are machine-level models, thus this section is devoted to such approaches.

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 utilisation [12]. Economu et al. [13] 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 utilisation, also the 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. Lim et al. [15] present a power budgeting system for virtualized infrastructures that enforces power limits on individual distributed applications. The authors utilise the method proposed by Kansal et al. to budget power for VM of virtual clusters in data centers. Bohra et al. in [16] proposed *vMeter*, a power modelling technique. The authors observed a correlation between the total system's power consumption and component utilisation. They proposed a four-dimensional linear weighted power model for the total power consumed. The components of the model are:

DOI: 10.1002/cpe

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. [17] 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 behaviour of a single work node and includes the contribution from individual components. Liu et al. [18] proposed a GreenCloud architecture that utilises live migration of VMs based on power information of the physical nodes reducing energy consumption for applications running in clouds, specifically for online gaming. Chengjian et al. [19] present a system power estimation and VM power metering by using performance events counter. The power models are built to infer power consumption from the system resource usage such as CPU and memory which can be indicated by certain performance events counter value. Stoess et al. [20] consider a prototype implementation targeting hypervisor-based virtual machine systems. The authors develop a framework for energy management in modular, multi-layered operating system structures. The framework explicitly takes the recursive energy consumption into account, which is spent in the virtualized layer or subsequent driver components. The framework provides a unified model to partition and distribute energy, and mechanisms for energy-aware resource accounting and allocation. Kim et al. [21] suggest a model to estimate the energy consumption of VMs. The model estimates the energy consumption of a VM based on the in-processor events generated by the VM. Based on the estimation model, the authors propose the energy-credit scheduler, a VM scheduling algorithm that conforms to the energy budget of each VM machine. Li et al. [22] provide an online power metering model at per VM level using a nonintrusive approach. Firstly, the characteristics are analysed and a basic ternary linear regression model is proposed based on the performance of three major components impacting VM power: CPU, memory and disk. Based on the comparison of estimated power values and measured values in experiments, inadequacies and improvements of the basic model are found and the sub classified piecewise linear model is proposed to improve the basic model. The model is used to estimate the power consumption of a physical server as well as one or more VMs running on it. Basmadjian and de Meer presents a model of multi-core processors [23], underlying differences between multi-core and single-core system power consumption modeling.

Linear regression model is frequently used to model power consumption and performance indicators. Bellosa [24] verified the linear relationship between power consumption of processor and some performance counters. Multivariate linear regression is also used to build models between power consumption and metrics of multiple components [12, 13, 25, 26, 27]. The further elaboration of such models includes usage of techniques such as clustering and decision trees, to create models applicable for wide range of applications [28].

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 [29]. 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 [7]. 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.

DOI: 10.1002/cpe

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 [1, 2].

The first contribution of the proposed model is an extension of the classical definition of resources. Instead of representing resources as discrete entities, the holistic model represents each resource by a resource vector: $res = (typ_1, \ldots, typ_z)$. The vector is composed of *resource types* that represents distinct resource types, e.g. Computing, Memory, Storage, and Network. Every resource type is further expressed as a vector of *resource supplies*: $typ_i = (sup_{i1}, \ldots, sup_{iy})$ that represents the exact implementation and number of resource supplies, i.e. hardware components. Finally, every 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, disk size, or bandwidth). 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 Figure 1. The node is composed of four resource types. Each of the types is composed of specific supplies: in this case there are 4 symmetric cores of CPU, single supplies for Memory and Storage, and 2 Network interfaces. The architecture ordering could be done based on the comparison of architectures of this node with other architectures in the same data centre.

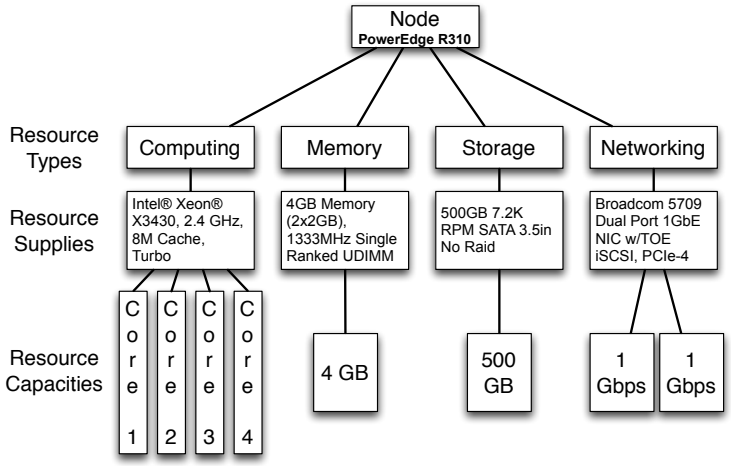

Figure 1. Example of a representation of a computing node within an HPC cluster

The second contribution is the addition of a machine configuration layer that corresponds to the used middleware (e.g. Figure 2). 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 utilisation of its components or environmental factors (e.g. temperature, supply voltage). In this work we take into account utilisation 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.

DOI: 10.1002/cpe

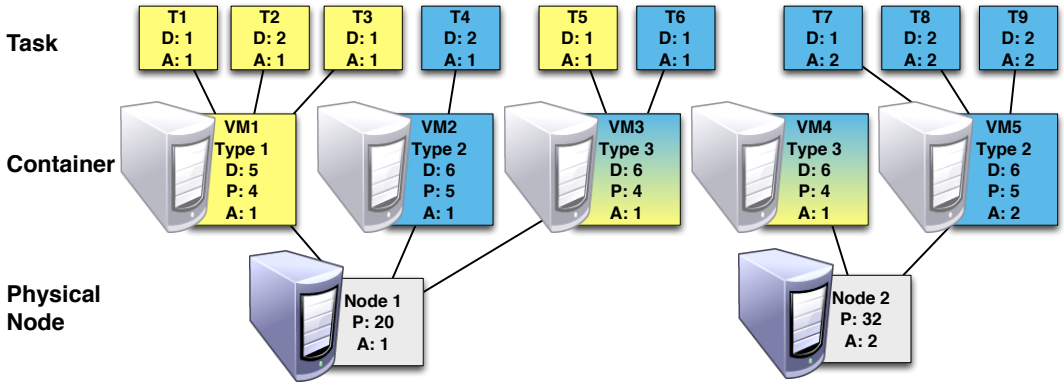

Figure 2. Example of allocation for one resource-type machines.

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 providers are the resources of 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 2 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. D denotes the resource demand of resource consumers. 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 utilisation with the obtained performance and power. In this paper, we present two approach for power modelling: multiple linear regressions and neural networks. Both methods can be 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 characterise. The next section details the experimental setup used 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. Table III provides an overview of the selected systems we compare in this article.

The benchmarking workflow, as presented in Figure 3, has been described in the following paragraphs. The baseline benchmark uses a customised 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

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

Vendor	Site	Cluster	#RAM	Processors		R_{peak}
Intel	Lyon	Taurus	32GB	2 Intel Xeon E5-2630@2.3GHz	12C	110,4 GFlops
AMD	Reims	StRemi	48GB	2 AMD Opteron 6164 HE@1.7GHz	24C	163.2 GFlops

DOI: 10.1002/cpe

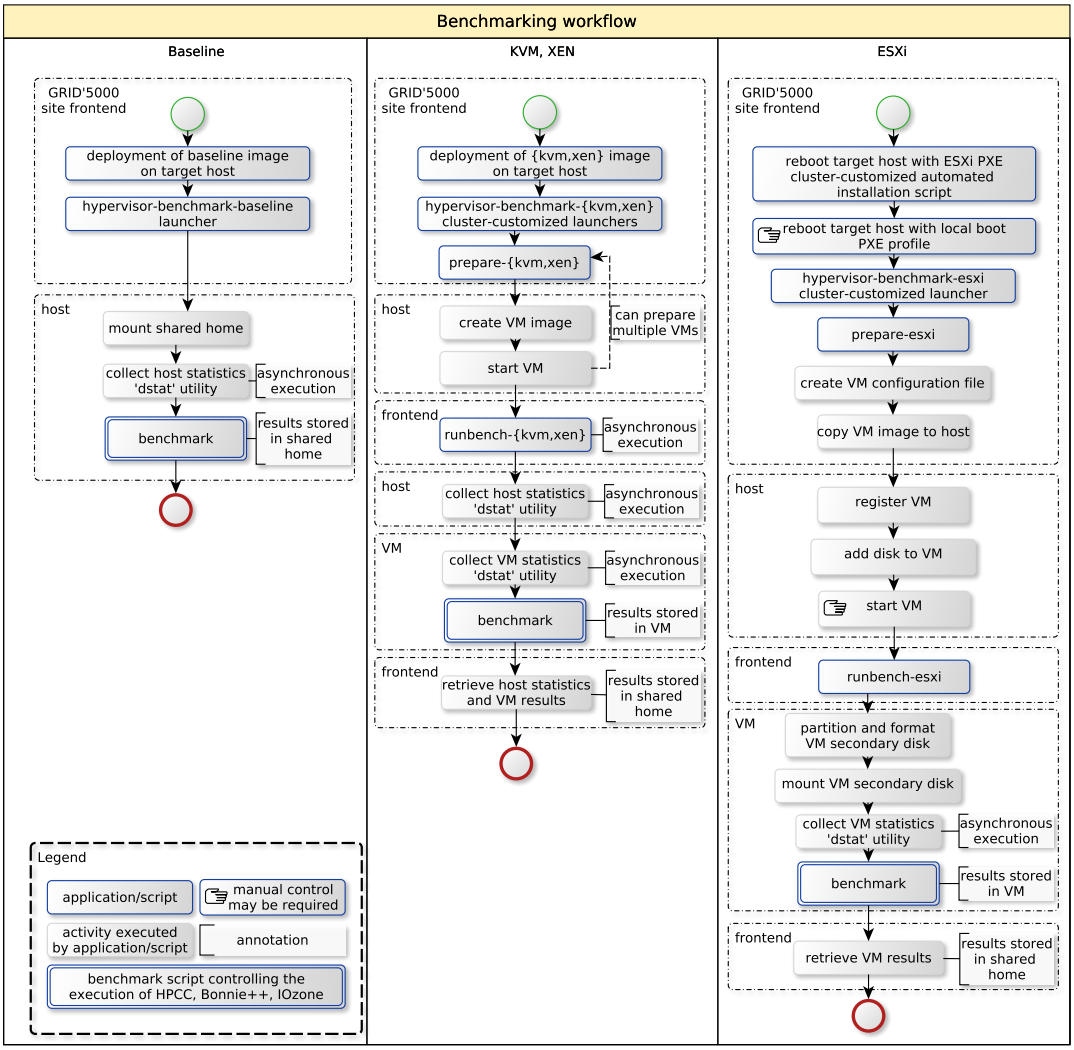

Figure 3. Benchmarking workflow.

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 with the output from the dstat utility. The archive is placed directly in the user's NFS shared home.

The benchmarking workflow for KVM and XEN is identical, although it is based on different scripts customised to work with these hypervisors, deploying KVM or Xen - enabled system images. 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

DOI: 10.1002/cpe

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-customised 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 was as for the baseline, KVM and XEN tests. Table IV presents the number of experimental runs for combinations of environments and nodes. Due to the need of manual interventions and special preparation of cluster, the ESXi environment was tested only on nodes 7, 9, and 10 in Taurus cluster and nodes 30 and 31 in StRemi cluster. The baseline environment was tested only 4 times on Taurus-8 due to technical problems with that node that appeared at the end of the sequence of experiments.

Table IV. 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	6439
taurus-8	4	5	5	0	4769
taurus-9	5	5	5	5	6285
taurus-10	5	5	5	5	6545

To accurately estimate the status of the analysed 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 resolution of 7W and ISO/IEC $\pm 1\%$ billing-grade accuracy (see [27] for discussion about using the Raritan meters), 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 utilisation metrics of CPU, memory, disk I/O and networking I/O 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

[‡]http://www.raritan.com

DOI: 10.1002/cpe

readings are possible. In order to prepare the data for modeling, the dstat values are aggregated (summed for flow values such as I/O, averaged for utilisation 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 are discarded from modeling (the amount of data removed in this way is 1.1%, 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 categorical 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. The only outlier removal procedure is pruning the rare occurrences of infeasible power readings: the observations that include less than 45W of measured power are removed. The 45W threshold is a conservative value chosen to be less than the half of the normal power of an idle machine. As a result, 73593 observations are used in further studies, as presented in detail in Table IV.

5. EXPERIMENTAL RESULTS

The presentation of the experimental results is divided into three parts. Section 5.1 presents the raw performance scores.

Section 5.2 relates the scores with the energy consumption of various hypervisors, to estimate energy efficiency of various configurations. Finally, Section 5.3 exhibits the results of modelling by multiple linear regressions and neural networks, together with the formal analysis of the quality of the models, and sample prediction plots.

5.1. Performance analysis

While the core 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, STREAM and RandomAccess. First of all, a synthetic view indexed over the hardware architecture is proposed in Figure 4. 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 can be seen in these plots that the three considered hypervisors present varying overhead as regards the computing benchmarks. In the computationally intensive HPL and DGEMM phases, the hypervisors show a limited (up to 4%) overhead on the Intel-based platform. On the AMD-based platform KVM and ESXi show a 7% overhead running the HPL benchmark and of 4% in DGEMM. Xen performs significantly worse in these tests on AMD, showing around 50% and 30% overhead respectively in these benchmarks. The memory-oriented RandomAccess test reveals a large overhead for the KVM on both hardware platforms and limited performance degradation for Xen on AMD. Interestingly, in the STREAM test on both platforms, some virtualization performance values are higher than the corresponding baseline scores, showing the effect of hypervisor caching mechanisms. This behaviour is further discussed in Section 5.2.

One inherent limitation to the usage of virtualization in an HPC environment obviously resides in the huge overhead induced on I/O operations. Thus, we present results of the IOZone benchmark in Figure 5, corresponding to different usage scenarios. Typically a significant degradation of the performances is observed as soon as a hypervisor is present, however there is a surprising element as regards the `read`, `reread` and `random_read` test on the ESXi environment which performs

[§]http://www.r-project.org

DOI: 10.1002/cpe

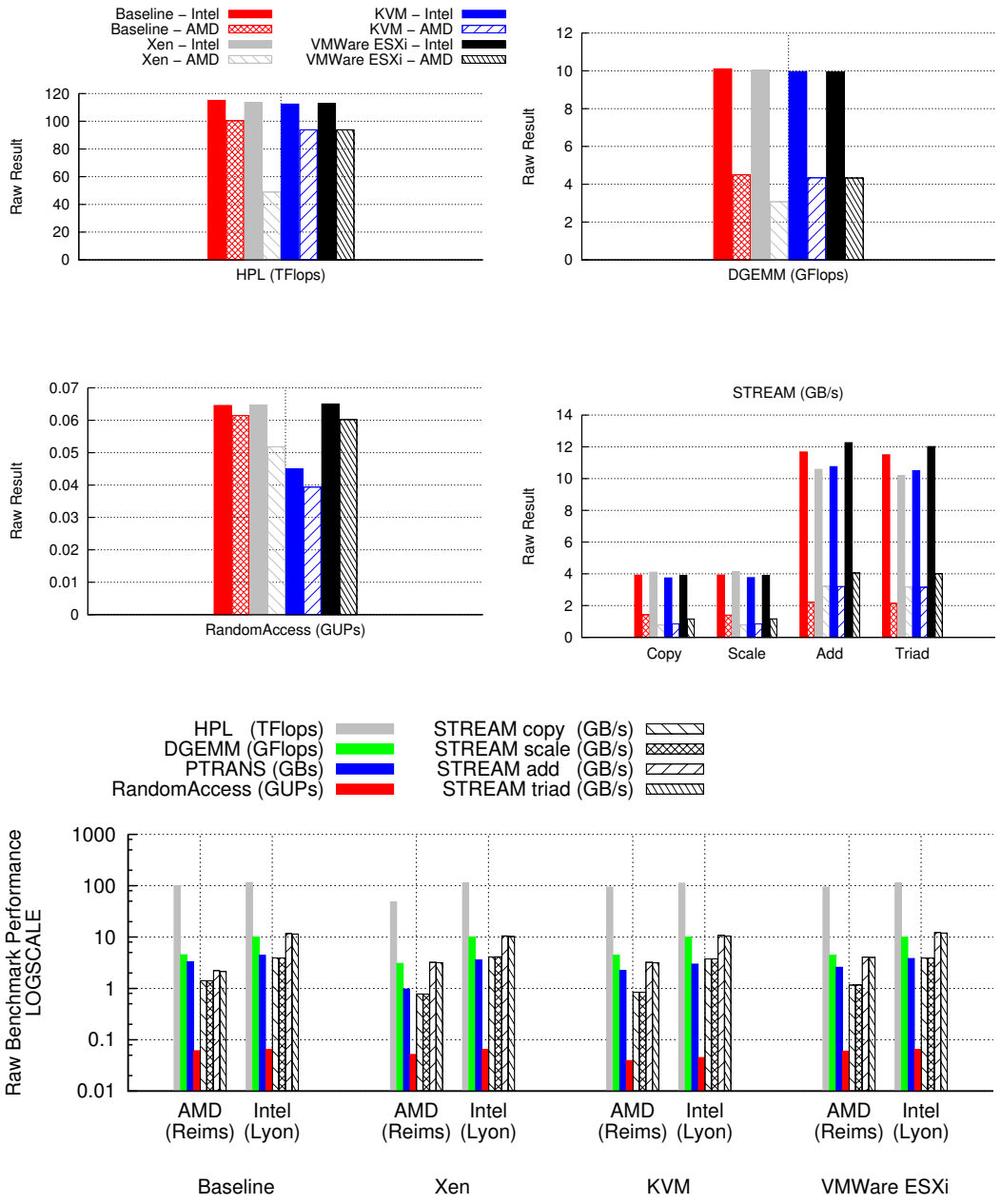

Figure 4. Average performance of the considered virtualization technologies.

better than the bare-metal system in the 1GB file benchmark. This is probably due to a favorable caching strategy on the file system deployed by this environment.

5.2. *Energy-efficiency Analysis*

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 Figure 6 traces of selected runs, and in Figure 7 a comparison across the considered hypervisors of the Performance per Watt (PpW) obtained during the HPCC tests detailed in Section 2.2. The figure presents some trends in the energy consumption of hypervisors. The baseline environment is shown to be superior

DOI: 10.1002/cpe

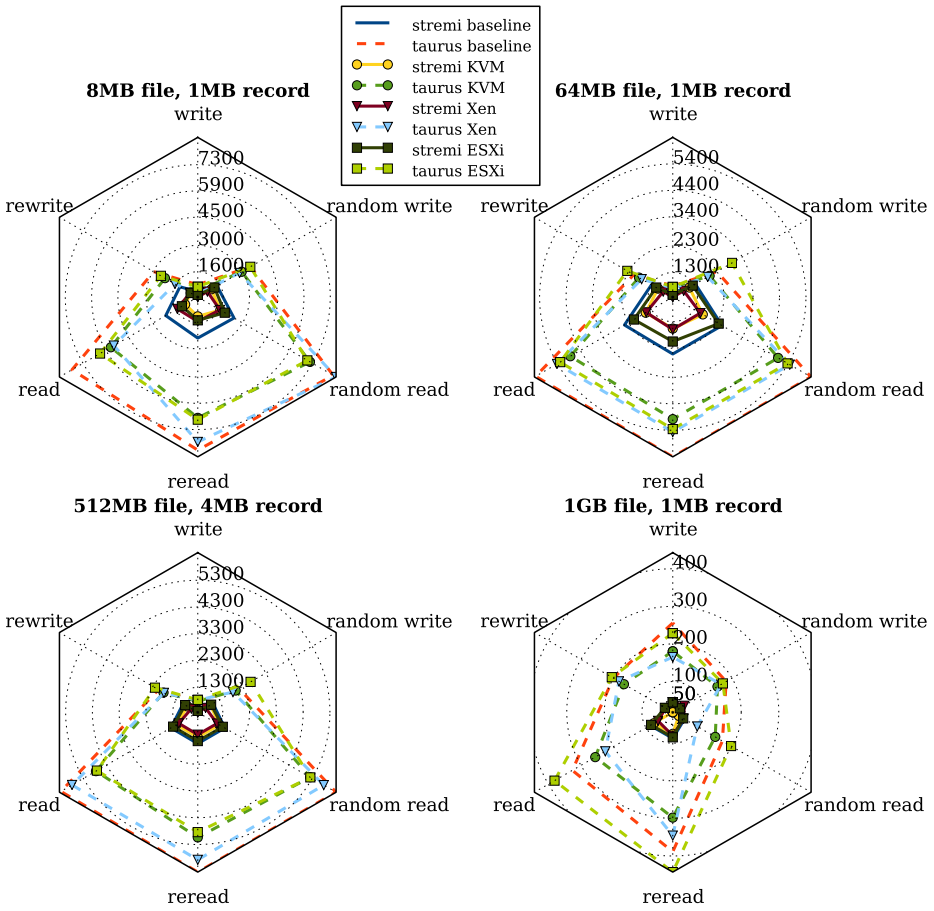

Figure 5. IOZone results for the selected hypervisors with different I/O usage scenarios.

except in some cases of the STREAM benchmark. The STREAM tests measure memory bandwidth, exhibiting high spatial locality, thus caching and prefetching mechanisms in the hypervisors allow the VMs to attain better performance (resulting also in higher PpW) than the baseline environment, behaviour which has also been observed in other studies [30]. In the CPU-intensive phases of HPCC (HPL, DGEMM), the Xen hypervisor performs best on the Intel-based Taurus cluster, and worst on the AMD-based StRemi cluster – where KVM has the highest PpW, scoring slightly better than ESXi. In both clusters on the FFT tests, whose memory access patterns exhibit high temporal locality, the highest relative PpW results are obtained by the ESXi hypervisor, and the worst by Xen. ESXi has the best PpW on the AMD platform, while on Intel Xen performs slightly better in the HPL, DGEMM and RandomAccess tests. The ESXi host OS is lightweight compared to Xen and KVM which run multiple Linux system-level daemons. Its low OS overhead enables it to achieve similar or better results in most tests, and appears to also take better advantage of the AMD-V virtualization extensions than the other hypervisors.

5.3. Power Modelling

The analysis of the traces left of each run on the selected configuration permitted to refine the parameters of the holistic model presented in Section 3. To ensure fair comparison of various models, we perform the modelling on the identical input data. The observations prepared as described in Section 4 are divided into training and test sets. The R^2 and Residuals statistic presented

DOI: 10.1002/cpe

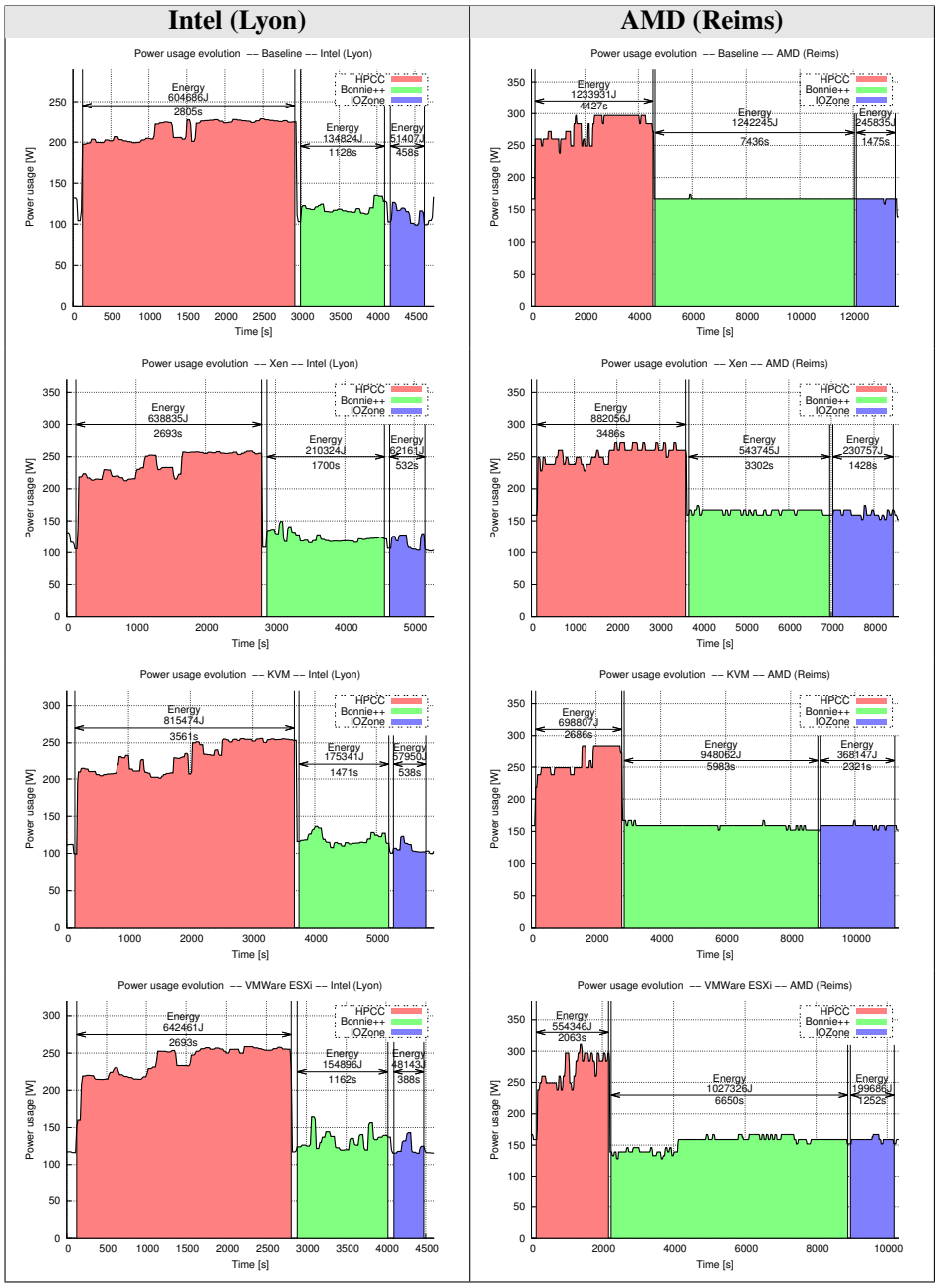

Figure 6. Power profile of selected runs in each configuration.

in this section are generated using training set, while the error statistics are computed for the test set.

5.3.1. Multiple Regressions The presented approach to model power using utilisation metrics as predictors is multiple regression. This method's advantage is the low computational complexity, no need for parameters and deterministic results. The final model is presented as a linear function of predictors [31]:

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

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 $(\beta_1,\ldots,\beta_k)$

DOI: 10.1002/cpe

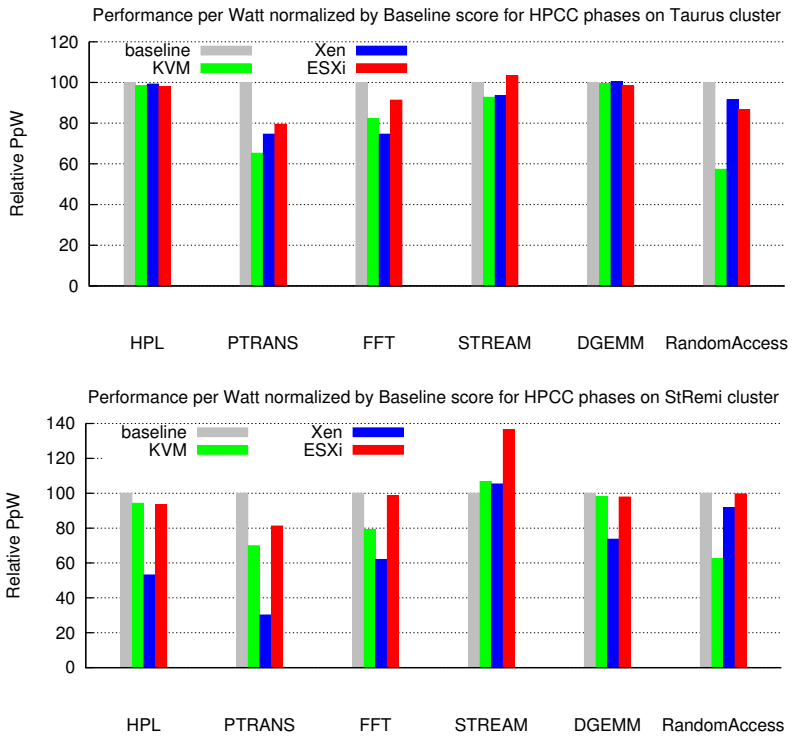

Figure 7. Comparison of the relative Performance per Watt for individual HPCC benchmarks

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, i.e. benchmark phase, hypervisor type, and node/cluster name. The aim of modelling at the node level is twofold: to analyse the operation of a system and to predict its behaviour. The results of multiple regressions are presented according to concepts of *complete-pooling* and *no-pooling* [32]. 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 (possible only in Complete-pooling analisys)
8. *Group Only* – all possible categorical predictors, no numerical predictors

Two distinct division methods were used to divide data to training and test sets. The first approach is random sampling: 2/3 of the observations are sampled as a training set, while the rest form a test set; it is further referred to as random sampling. The second method divides the data by full runs: the training set consist of 3 runs of each configuration while the test set consist of 2 runs (or 1 in case only 4 runs were successfully performed) and is referred to as sequential division. The comparison

DOI: 10.1002/cpe

of the two methods is designed to compare the method that uniformly samples the whole data set with more realistic assumption that after gathering of training set data, the model is continuously used in production scenario, represented by the runs from the test set.

The tables in this section are based on various quality indicators of models. R^2 value, distribution of residuals including standard error, minimal and maximal values, first and third quartile, and median are based on the training set. The test set was used to calculate the prediction errors, defined as difference between power values observed during experiments and the ones predicted by a model. The average values of absolute prediction error (|Error|) and average values of prediction error (Error) are presented in Watts or as relative error in percents. The absolute prediction error can be treated as a value indicating a temporary accuracy of a model (it presents how far is on average the prediction from the measured value), while the prediction error can be treated as the indicator of long-run accuracy model (i.e. to what extent errors are cumulative).

The results of Complete-pooling analysis are presented in Table V for random sampling and in Table VII for the sequential division. The model fitting quality indicators are similar in both cases of data set division. The models that take into account all predictors (*Basic* and *Refined*) have the highest R^2 values, the smallest residual standard 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, which however presents an unexpected property of cancelling large temporal errors in long run, represented by the best average prediction error among all models in sequential division case. 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, 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 .

Contrary to the above mentioned similarity, there is a significant degradation in values of average errors in case of sequential division, which points that random sampling results in over fitting the data and this cannot be used to assess the final accuracy of the model. Despite the larger values of errors, the sequential division achieves smaller relative errors, meaning that the large prediction errors occur more often for large observed values.

The quality of No-pooling models is presented for clusters StRemi and Taurus in Table VI for random sampling and in Table VIII for the sequential division. 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 motivated by the differences in the underlying hardware. As a result, No-pooling models should have better quality than corresponding Complete-pooling models. Despite that, the quality of No-pooling modelling is apparently worse than the complete-pooling model, especially in the case of sequential division, but it must be taken into account that the averaging results from both clusters in complete-pooling case can result in decreasing overestimation error in one cluster by underestimation in the other. Additionally, the longer running times of experiments on StRemi nodes resulted in more observations from this cluster, that bias the Complete-pooling model results and 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. The results of error prediction analysis in Taurus cluster suggests larger variability in the data for this cluster, as the prediction errors are large for three complex models, listed in the Table VIII as the first three.

The results of models can be used to create an instance of the 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 the holistic model for decision making is presented in this

DOI: 10.1002/cpe

Table V. Complete-pooling models quality, random sampling

Model	R^2	Residuals						|Error|		Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%	W	%
Basic	0.957	10.7	-119	-3.46	0.921	5.1	116	6.85	3.8%	0.0025	0.4%
Refined	0.957	10.7	-119	-3.46	0.921	5.1	116	6.85	3.8%	0.0025	0.4%
No Phases	0.938	12.8	-130	-4.56	0.72	5.14	125	8.32	4.4%	-0.022	0.5%
CPU Hom.	0.819	21.9	-147	-12.7	3.71	13.6	158	16.5	9.4%	-0.042	1.5%
CPU Het.	0.922	14.3	-134	-5.63	0.28	5.05	127	9.86	5.1%	-0.03	0.5%
No group	0.86	19.2	-142	-8.08	3	11.2	129	13.8	7.8%	-0.027	1.1%
Clusterwise	0.927	13.9	-122	-7.68	1.17	7.98	132	10.2	5.5%	0.059	0.6%
Group only	0.926	14	-116	-4.08	1.83	5.58	90.8	8.58	4.8%	-0.12	0.6%

Table VI. No-pooling models quality, random sampling

No-pooling StRemi cluster models quality

Model	R^2	Residuals						|Error|		Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%	W	%
Basic	0.968	8.78	-104	-2.86	0.447	3.54	99.9	5.24	2.7%	0.022	0.2%
Refined	0.968	8.78	-104	-2.82	0.448	3.54	99.9	5.24	2.7%	0.022	0.2%
No Phases	0.954	10.5	-113	-3.23	0.797	3.49	118	6.08	3.0%	-0.033	0.2%
CPU Hom.	0.925	13.4	-116	-7.12	0.0869	7.94	122	9.35	4.7%	-0.012	0.4%
CPU Het.	0.937	12.2	-119	-4.3	1.01	3.93	122	7.7	3.7%	-0.1	0.3%
No group	0.942	11.8	-120	-6.35	1.06	6.77	115	8.13	4.2%	0.046	0.4%
Group only	0.96	9.76	-103	-3.09	0.774	4.31	112	5.87	3.0%	-0.0068	0.2%

No-pooling Taurus cluster models quality

Model	R^2	Residuals						|Error|		Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%	W	%
Basic	0.956	11.8	-116	-5.67	0.0885	5.7	130	7.74	4.7%	-0.033	0.4%
Refined	0.956	11.8	-117	-5.67	0.0762	5.7	130	7.74	4.7%	-0.032	0.4%
No Phases	0.924	15.6	-129	-9.06	-0.214	11	144	11.6	6.5%	0.041	0.7%
CPU Hom.	0.897	18.1	-129	-11	-1.62	15.8	136	14	7.6%	0.2	1.0%
CPU Het.	0.902	17.7	-136	-11.2	-1.58	16.2	132	13.8	7.6%	0.16	1.0%
No group	0.916	16.4	-128	-9.25	-0.262	12.2	147	12.4	6.8%	0.15	0.8%
Group only	0.905	17.3	-114	-5.85	0.834	7.45	109	10.4	6.5%	-0.27	0.9%

section. Table IX 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 in Taurus node it is up to 10W.

The phases also have a large impact on the final power consumption in the derived model (Table IX, second row). The presented values are *adjustments* to the amount based on the utilisation metrics. Therefore, there is no sense in the direct comparison of these values (e.g. nodes do not consume approximately 2.8W *more* in an idle state than during the STREAM phase). However, these values show that knowledge about the running application's characteristics can add valuable information to power predictions. The influence of the hypervisor type 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 modelling. Finally, we discuss the numerical predictors. The intercept has a high positive value. The CPU utilisation 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.

DOI: 10.1002/cpe

Table VII. Complete-pooling models quality, sequential division

Model	R^2	Residuals						|Error|		Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%	W	%
Basic	0.958	10.5	-113	-3.39	0.547	5.09	115	7.81	4.4%	-0.67	0.1%
Refined	0.958	10.5	-113	-3.39	0.547	5.09	115	7.81	4.4%	-0.67	0.1%
No Phases	0.94	12.6	-117	-4.49	0.551	4.94	126	9.52	5.2%	-1	-0.1%
CPU Hom.	0.817	21.9	-147	-12.1	3.89	13.4	161	16.4	9.4%	-0.21	1.5%
CPU Het.	0.923	14.2	-126	-5.14	-0.298	5.34	125	10.3	5.3%	0.44	0.9%
No group	0.858	19.3	-142	-8.14	2.95	10.7	123	14.7	8.4%	-1.3	0.5%
Clusterwise	0.928	13.8	-123	-7.52	0.908	7.25	130	11.6	6.4%	-1.1	-0.0%
Group only	0.926	14	-112	-3.51	1.63	5.15	90.3	8.63	5.0%	0.9	1.2%

Table VIII. No-pooling models quality, sequential division

No-pooling StRemi cluster models quality

Model	R^2	Residuals						|Error|		Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%	W	%
Basic	0.972	8.16	-103	-2.8	0.0674	3.08	106	5.8	3.1%	0.75	0.8%
Refined	0.972	8.16	-103	-2.8	0.0736	3.08	106	5.8	3.1%	0.73	0.8%
No Phases	0.957	10	-114	-2.95	0.0543	3.42	123	6.6	3.4%	0.72	0.8%
CPU Hom.	0.929	12.9	-116	-7.51	-0.326	7.37	128	10.1	5.2%	0.88	1.0%
CPU Het.	0.941	11.8	-119	-3.71	0.555	4.05	125	8.34	4.1%	0.73	0.9%
No group	0.946	11.3	-120	-6.57	0.736	6.84	127	8.71	4.6%	0.87	1.0%
Group only	0.965	9.12	-103	-3.13	0.126	3.89	112	6.38	3.4%	0.67	0.8%

No-pooling Taurus cluster models quality

Model	R^2	Residuals						|Error|		Error	
		St.er.	Min	1Q	Median	3Q	Max	W	%	W	%
Basic	0.955	11.9	-115	-5.91	0.126	5.81	124	11.9	7.1%	-5.1	-2.6%
Refined	0.955	11.9	-115	-5.91	0.138	5.82	124	11.9	7.1%	-5.1	-2.6%
No Phases	0.922	15.7	-120	-9.42	0.27	11	137	17.5	10.0%	-7.5	-3.8%
CPU Hom.	0.893	18.4	-130	-10.9	-1.77	15.9	135	13.9	7.6%	-0.27	0.8%
CPU Het.	0.899	17.9	-128	-11.4	-1.57	15.9	130	13.7	7.6%	-0.074	0.9%
No group	0.914	16.5	-117	-9.41	-0.556	12.2	148	16.6	9.3%	-5.4	-2.5%
Group only	0.901	17.7	-111	-5.64	1.03	7.57	108	9.99	6.4%	1.1	1.7%

Table IX. 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	-2.2	-14	-3.2	-45	-36	-46	-43

Bonnie	DGEMM	FFT	HPL	IOZONE	PTRANS	RandomAccess	STREAM	idle
0	12	14	22	-0.04	-0.36	-5.3	7.2	10

ESXi	KVM	Xen	baseline
0	-2.8	-4.1	-14

Intercept	cpu user	cpu system	cpu idle	cpu wio	mem used
314	-0.78	-1.2	-1.7	-1.8	8.9E-10

mem buffers	mem cached	mem free	disk write	disk read	bytes rec.	bytes send
6.7E-09	6.6E-10	5.9E-10	-6.3E-10	-3.7E-10	2.8E-05	-1.1E-03

5.4. Neural Network Modelling

The neural network modelling is a tool of artificial intelligence that can accurately describe or control complex systems, with little a priory theoretical knowledge. The goal of a neural network is to map a set of input patterns onto a corresponding set of output patterns. The network carries out the mapping by first learning from a series of past examples defining input and output sets for the given system. The network then applies what it has learned to a new input pattern to predict the appropriate output. Figure 8 shows a multilayer feed-forward neural network formed by an interconnection of

DOI: 10.1002/cpe

nodes. This neural network has an *input* layer, a *hidden* layer, and an *output* layer. A neural network can be considered as a black box into which an specific input is sent to each node in the input layer. The input layers receive the information from an external source, and send this information to the network for processing. Then, the network processes this information through the interconnections between nodes. The hidden nodes receive the information from the input layer, and process it. The entire processing step is hidden from view. Finally, the network provides an output from the nodes on the output layer. The output nodes receive the processed information from the network, and send the results out to an external receptor.

Some advantages of neural network models are: *automated abstraction* without needing an expert in a particular problem-solving domain to develop knowledge base that expert systems require, *the information is distributed over a field of nodes* providing greater flexibility than one finds in symbolic processing, *neural networks can learn*, that is, if an error or novel situation occurs that produces inaccurate system results, an error-correction training technique can be used to correct it by adjusting the strengths of the signals emitted from nodes until the output error disappears. Another advantage is that neural networks are better suited for processing incomplete, noisy or inconsistent data. Neural networks are used for classification, prediction, data conceptualization, data association, data filtering, and optimization.

In this study, we are interested in a feed-forward network. Multilayer feed-forward neural networks are the most popular and most widely used models in many practical applications. Multilayer neural networks can represent a large variety of problems and estimate non-linear effects, the study of neural networks is therefore an additional validation of the results of linear regressions. The data for the neural network training is additionally prepared: all numerical values are normalised by dividing them by the maximum value in the population, while categorical variables are coded using additional dummy variables for each value of a factor.

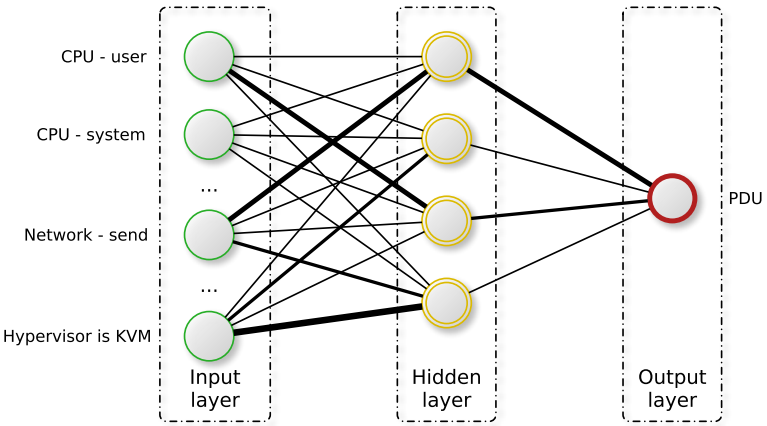

Figure 8. Neural Network design

The neural network is implemented in R, using the `nnet` package. The stopping criterion were 250 iterations of the network optimisation by quasi-Newton method (variable metric algorithm). The output units have logistic function and the network fitting is guided by a least-square function. Studied networks allow skip-layer connections (between input and output layers). The initial weights of the network are randomly initialised with values close to 0.5. 11 various network sizes are tested, including networks with 0 to 10 hidden units. As the training process is non-deterministic, 30 independent runs are performed for each number of hidden units. As the sequential division is more realistic and challenging for models, we test only this method of division data set for training of neural networks.

The aggregated results presented in Table X describe the average performance of the neural network models. The results of incorrectly fitted networks (defined as networks with $R^2 < 0.8$) were removed before aggregating the results of independent runs. All of the models with hidden neurones present good fitting to the training data, visible as high R^2 values. Adding additional

DOI: 10.1002/cpe

Table X. Neural networks models quality, sequential division

Hidden Units	R^2	Residuals					|Error|		Error	
		Min	1Q	Median	3Q	Max	W	%	W	%
0	0.884	-130	-9.01	2.12	10.2	207	16.4	8.6	-8.82	-3.45
1	0.949	-125	-5.8	0.784	5.96	129	12.1	6.21	-7.49	-3.06
2	0.96	-119	-4.96	0.686	5.22	125	11.8	6.07	-7.92	-3.44
3	0.964	-118	-4.43	0.7	4.84	127	11.8	6.08	-8.2	-3.59
4	0.967	-117	-3.96	0.753	4.56	125	11.6	5.98	-8.14	-3.61
5	0.968	-117	-3.83	0.772	4.51	126	11.5	5.89	-8.05	-3.55
6	0.969	-116	-3.73	0.798	4.38	124	11.7	6.03	-8.41	-3.78
7	0.97	-116	-3.57	0.786	4.39	126	11.8	6.09	-8.8	-4.03
8	0.971	-116	-3.55	0.81	4.33	123	11.8	6.12	-8.72	-3.98
9	0.971	-116	-3.49	0.793	4.31	125	11.8	6.11	-8.76	-4.01
10	0.971	-117	-3.44	0.799	4.28	124	11.8	6.1	-8.64	-3.92

hidden units increases the fit accuracy, as R^2 increases and the distribution of residuals is closer to 0. However, the increased fit does not necessarily result in better performance during validation by the test set: the average absolute error values is the lowest for 5 hidden unit while the average error is the lowest for a single hidden unit. The observation indicates that large neural network sizes can be counter-productive. On the other hand it proves, there are no significant non-linear effects that would be better mapped by more complex networks.

The analysis of the error values reveals bad running properties of the neural network modelling. While the absolute error values are still acceptable in comparison with the values of less performant linear regression models, the values of errors reveals that the prediction by neural network is underestimated by few Watts, which can lead to more significant cumulation of error for longer estimation periods.

5.5. Predicting Power using the Proposed Models

Sample results of power prediction by the Refined non-pooling models and the Neural Model identified as the best one among all runs are presented in Figure 9. The figure presents the predicted 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 the most 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.

6. CONCLUSION

In the paper we introduce and experimentally evaluate a holistic model of virtualized computing nodes and evaluate its applicability for power estimation. The principle of the model is based on division of a system into three layers: Physical Node (Server), Container (VM), and Task (Application), and further description of each of these layers in terms of distinct resource types. The utilisation of the resource types is combined with the discrete information about host, hypervisor, and application types to derive models of the power consumption of a system. The utilization data was preprocessed to represent an overhead of monitoring acceptable in production settings, e.g. all measurements corresponds to 15s intervals.

The parameters of the models have been refined by a set of concrete experiments on the Grid'5000 platform using three widespread hypervisors, namely Xen, KVM and VMware ESXi on the two leading hardware architectures (AMD and Intel).

DOI: 10.1002/cpe

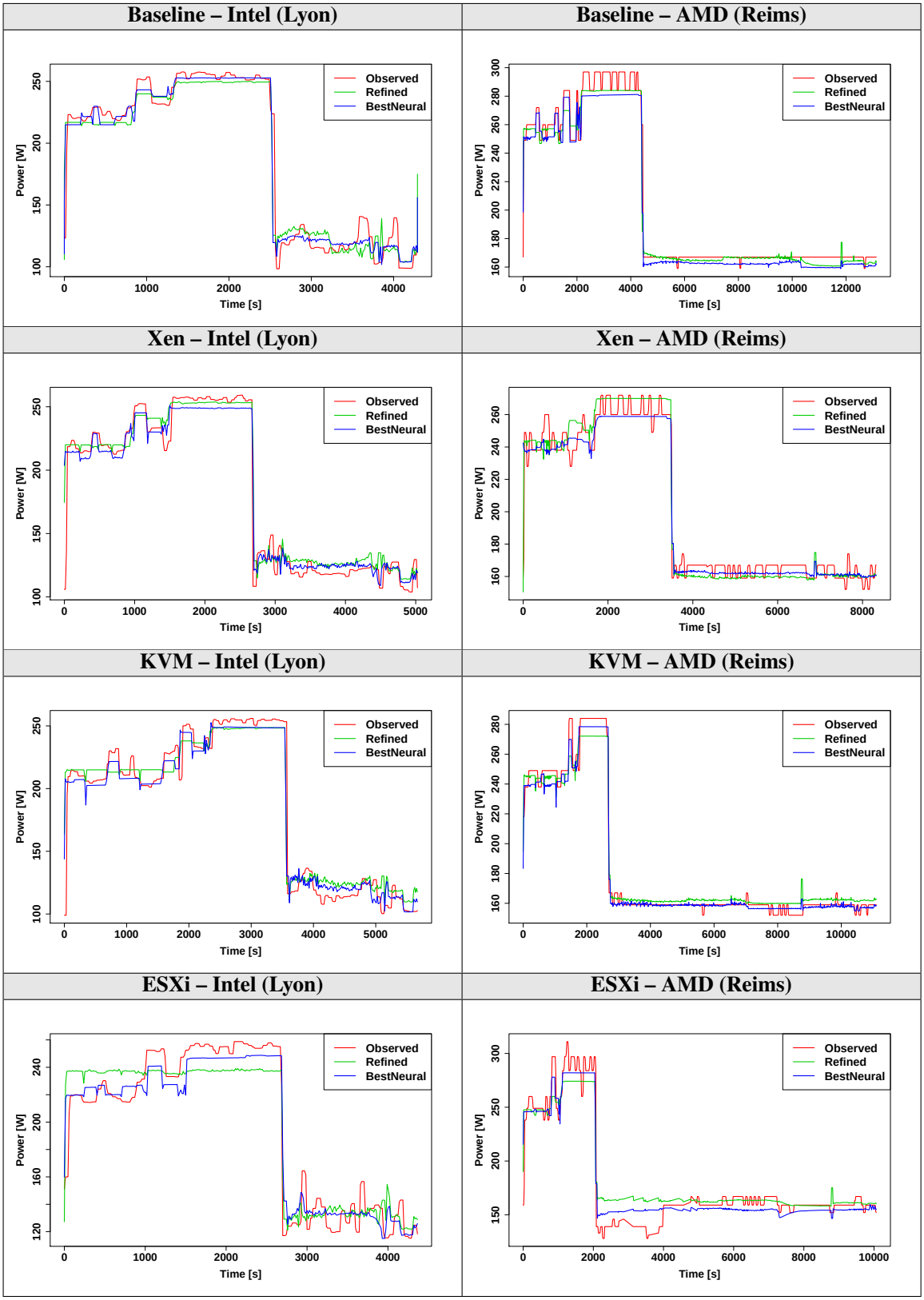

Figure 9. The predicted energy profiles for the best identified models.

DOI: 10.1002/cpe

When compared to a *baseline* environment running in native mode, the usage of hypervisors in an HPC environment raises a limited overhead, and can be foreseen if 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.

Two distinct modelling approaches are used in this study to finally derive power models: multiple linear regression and neural network.

The holistic power modelling using multiple linear regression is able to accurately estimate the power consumption in a virtualized system with an acceptable temporary and cumulative errors. The model is lightweight – it is represented by a single equation, and its creation 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 modelling based on the neural network results in larger errors and requires orders of magnitude more time for the training of the network, in comparison with linear regression. The risk of overfitting and the cost of training discourages usage of networks with more than 5 hidden units. The neural network models presented significantly larger cumulative errors. Modifying an existing network can require additional costly training phase. Larger error of neural network approach suggest that there are no significant non-linear behavior that could be exploited by non-linear activation function of neurones.

The high quality of the prediction with respect to related work despite low-frequency sampling of high-level predictors indicates that using wide range of metrics that covers not only CPU behaviour is sufficient to determine the power of a system, even in a virtualized system.

The future work includes adding more elements to the holistic model, 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 loads 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. This future work will require further experimentation on a larger set of applications and machines.

ACKNOWLEDGEMENTS

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 FNR INTER/CNRS/11/03 Green@Cloud project. M. Guzek acknowledges the support of the FNR in Luxembourg and Tri-ICT, with the AFR contract no. 1315254. 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. Blazewicz J, Ecker K, Schmidt G, Węglarz J. *Scheduling in Computer and Manufacturing Systems*. Springer-Verlag: New York, 1994.
2. Sinnen O. *Task Scheduling for Parallel Systems*. John Wiley & Sons: Hoboken, NJ, USA, 2007.
3. Barham P, Dragovic B, Fraser K, Hand S, Harris T, Ho A, Neugebauer R, Pratt I, Warfield A. Xen and the art of virtualization. *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP '03, ACM: New York, NY, USA, 2003; 164–177, doi:10.1145/945445.945462. Available online.
4. Kivity A, al. KVM: the Linux Virtual Machine monitor. *Ottawa Linux Symposium*, 2007; 225–230. URL http://www.kernel.org/doc/ols/2007/ols2007v1-pages-225-230.pdf.
5. Ali Q, Kiriansky V, Simons J, Zaroo P. Performance evaluation of hpc benchmarks on vmware's esxi server. *Proceedings of the 2011 international conference on Parallel Processing*, Euro-Par'11, Springer-Verlag: Berlin, Heidelberg, 2012; 213–222.
6. Grid'5000. [online] http://grid5000.fr.
7. Younge AJ, Henschel R, Brown J, von Laszewski G, Qiu J, Fox GC. Analysis of virtualization technologies for high performance computing environments. *The 4th International Conference on Cloud Computing (IEEE CLOUD 2011)*, IEEE, IEEE: Washington, DC, 2011.
8. J J Dongarra PL. Introduction to the hpcchallenge benchmark suite. *Technical Report*, ICL 2004.

9. Bonnie++. [online] http://www.coker.com.au/bonnie++/.
10. Iozone filesystem benchmark. [online] http://www.iozone.org/.
11. Costa GD, Hlavacs H, Hummel KA, Pierson JM. Modeling the energy consumption of distributed applications. *Handbook of Energy-Aware and Green Computing*, Ahmad I, Ranka S (eds.). Chapman and Hall CRC, 2012.
12. Fan X, Weber WD, Barroso LA. Power provisioning for a warehouse-sized computer. *Proceedings of the 34th annual international symposium on Computer architecture*, ISCA '07, ACM: New York, NY, USA, 2007; 13–23.
13. Economou D, Rivoire S, Kozyrakis C. Full-system power analysis and modeling for server environments. *In Workshop on Modeling Benchmarking and Simulation (MOBS)*, 2006.
14. Kansal A, Zhao F, Liu J, Kothari N, Bhattacharya AA. Virtual machine power metering and provisioning. *Proceedings of the 1st ACM symposium on Cloud computing*, SoCC '10, ACM: New York, NY, USA, 2010; 39–50.
15. Lim H, Kansal A, Liu J. Power budgeting for virtualized data centers. *Proceedings of the 2011 USENIX conference on USENIX annual technical conference*, USENIXATC'11, USENIX Association: Berkeley, CA, USA, 2011; 5–5.
16. Bohra A, Chaudhary V. Vmeter: Power modelling for virtualized clouds. *Parallel Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on*, 2010; 1 –8.
17. Chen Q, Grosso P, Veldt Kvd, Laat Cd, Hofman R, Bal H. Profiling energy consumption of vms for green cloud computing. *Proceedings of the 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, DASC '11, IEEE Computer Society: Washington, DC, USA, 2011; 768–775.
18. Liu L, Wang H, Liu X, Jin X, He WB, Wang QB, Chen Y. Greencloud: a new architecture for green data center. *Proceedings of the 6th international conference industry session on Autonomic computing and communications industry session*, ICAC-INDST '09, ACM: New York, NY, USA, 2009; 29–38.
19. Chengjian W, Xiang L, Yang Y, Ni F, Mu Y. System power model and virtual machine power metering for cloud computing pricing. *Intelligent System Design and Engineering Applications (ISDEA), 2013 Third International Conference on*, 2013; 1379–1382.
20. Stoess J, Lang C, Bellosa F. Energy management for hypervisor-based virtual machines. *2007 USENIX Annual Technical Conference on Proceedings of the USENIX Annual Technical Conference*, ATC'07, USENIX Association: Berkeley, CA, USA, 2007; 1:1–1:14.
21. Kim N, Cho J, Seo E. Energy-based accounting and scheduling of virtual machines in a cloud system. *Green Computing and Communications (GreenCom), 2011 IEEE/ACM International Conference on*, 2011; 176–181.
22. Li Y, Wang Y, Yin B, Guan L. An online power metering model for cloud environment. *Network Computing and Applications (NCA), 2012 11th IEEE International Symposium on*, 2012; 175–180.
23. Basmadjian R, de Meer H. Evaluating and modeling power consumption of multi-core processors. *Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet*, e-Energy '12, ACM: New York, NY, USA, 2012; 12:1–12:10.
24. Bellosa F. The benefits of event: driven energy accounting in power-sensitive systems. *Proceedings of the 9th workshop on ACM SIGOPS European workshop: beyond the PC: new challenges for the operating system*, EW 9, ACM: New York, NY, USA, 2000; 37–42.
25. Contreras G, Martonosi M. Power prediction for intel xscale processors using performance monitoring unit events. *Proceedings of the 2005 international symposium on Low power electronics and design*, ISLPED '05, ACM: New York, NY, USA, 2005; 221–226.
26. Lewis A, Ghosh S, Tzeng NF. Run-time energy consumption estimation based on workload in server systems. *Proceedings of the 2008 conference on Power aware computing and systems*, HotPower'08, USENIX Association: Berkeley, CA, USA, 2008; 17–21.
27. Witkowski M, Oleksiak A, Piontek T, Węglarz J. Practical power consumption estimation for real life {HPC} applications. *Future Generation Computer Systems* 2013; **29**(1):208 – 217. Including Special section: AIRCC-NetCoM 2009 and Special section: Clouds and Service-Oriented Architectures.
28. Jarus M, Oleksiak A, Piontek T, Węglarz J. Runtime power usage estimation of {HPC} servers for various classes of real-life applications. *Future Generation Computer Systems* 2013; (0):–.
29. Gavrilovska A, Kumar S, Raj H, Schwan K, Gupta V, Nathuji R, Niranjan R, Ranadive A, Saraiya P. High-Performance Hypervisor Architectures: Virtualization in HPC Systems. *Proc. of HPCVirt 2007*, Portugal, 2007.
30. HPC Application Performance on ESX 4.1: Stream. Available online.
31. Fox J, Weisberg S. *An R Companion to Applied Regression*. SAGE Publicatotions: Los Angeles, 2011.
32. Gelman A, Hill J. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press: Cambridge, 2009.

DOI: 10.1002/cpe