Deliverable D2.2

Green Management of Data Centres: model for energy and ecological efficiency assessment

Authors:

Mario Macías, Mauro Canuto, David Ortiz, Jordi Guitart (BSC)

The project *Advanced concepts and tools for renewable energy supply of IT Data Centres* has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 608679

Project acronym	RenewIT
Project number	FP7 – SMARTCITIES – 2013 - 608679
Project title	Advanced concepts and tools for renewable energy supply of IT Data Centres
Website	www.renewit-project.eu

Deliverable	D2.2
Title of deliverable	Green Management of Data Centres: model for energy and ecological efficiency assessment
Workpackage	WP2
Dissemination level	Public
Data of deliverable	31/06/2015
Authors	Mario Macías, Mauro Canuto, David Ortiz, Jordi Guitart
Contributors	Davide Nardi Cesarini (AEA), Eduard Oro (IREC)
Reviewers	Eduard Oro (IREC), Massimiliano Manca, Daniela Isidori (AEA),

Version	Date	Author	Organisation	Comments
0.1	31/10/2014	Mario Macias, Mauro Canuto, David Ortiz, Jordi Guitart	BSC	First version for midterm internal review
0.2	5/6/2015	Mario Macias, Mauro Canuto, David Ortiz, Jordi Guitart	BSC	First draft of the final version
1.0	26/6/2015	Mario Macias, Mauro Canuto, David Ortiz, Jordi Guitart	BSC	Final version, reviewed and ready to be submitted

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF TABLES

KEYWORDS

Service consolidation

Data centre

Ecological efficiency assessment

Energy efficiency assessment

Metric

Power Models

Virtual Machine

Virtualisation

EXECUTIVE SUMMARY

The flexibility of Cloud Computing as a computing paradigm allows its users to submit heterogeneous workloads that may perform an intensive use of diverse system resources: CPU, disk, memory, network… In addition to efficient buildings and hardware design, the high energy costs of large Data Centres require enabling policies to perform efficient operation also from the middleware perspective. This deliverable considers efficiency in terms of energy, as the ratio of useful work and the energy consumed to do it; future deliverables will also consider ecological efficiency, as the ratio of useful work and the ecological impact (for example, carbon emissions) to do it.

To enable efficient middleware operation, this deliverable is divided into two main blocks: (1) development of power models for energy assessment, (2) establishment of energy-aware policies to optimise the allocation and management of Virtual Machines (VM).

The released power models must be able to model heterogeneous workloads (Web, High-Performance Computing and Data-Intensive) that are executed in VM over the same hardware nodes. Power models are built following an iterative workflow: (1) to execute micro-benchmarks to stress individual system-level resources and model each part of the system; (2) to select which system-level resources have a true influence in the overall power consumption, and build a composite power model; (3) to characterise real application workloads to extract patterns of their usage of system-level resources; and (4) to validate the composite model by comparing the generalist power model predictions with the power that is measured during the execution of the workloads.

The experiments show that the power varies according to the intensity of usage of some resources such as CPU, cache and main memory, or network; other resources, such as rotational disks, have almost-constant power consumption, independently of the intensity of their usage. After all the executions and data extraction, a Neural Network algorithm is trained to model the power consumption of a workload as a function of their resources consumption. Models for both physical hosts and VM are released.

This deliverable releases a basic energy-aware resource manager. When a VM is deployed or un-deployed, or periodically, the system must enable policies to maximize the energy efficiency by means of consolidation, while the system provides the Quality of Service that is required by the workloads.

LIST OF ABBREVIATIONS

CPU	Central Processing Unit
FLOPS	Floating-point Operations per Second
HPC	High Performance Computing
I/O	Input / Output
KPI	Key Performance Indicators
KVM	Kernel-based Virtual Machine
MAPE	Mean Average Percent Error
PDU	Power Distribution Unit
PSU	Power Supply Unit
SLA	Service Level Agreement
SWaP	Space, Watts and Performance
VM	Virtual Machine
WP	Work Package

1 INTRODUCTION

Traditionally, academic and scientific entities as well as some companies owned big mainframes that had to be shared by their users to satisfy their computing requirements. These systems were managed centrally, considering performance metrics: throughput, response time, load-balancing, etc. The big mainframes paradigm [1] is transiting to a Cloud Computing-driven paradigm [2], where users do not own their resources and pay for the usage of remote resources. The main advantage is that users do not require spending neither an initial expenditure nor maintenance costs for the hardware, and pay only for the capacity that they are using in each moment.

Cloud Computing relies on Virtualisation [3] as a core technology that simultaneously executes full operating systems as guests within a single hardware node. RenewIT project models virtualised hosts because virtualisation brings the following advantages for both Clients and Cloud Providers:

1. Physical hosts can be shared transparently to the clients, which are isolated as if each client were using a dedicated physical host. Virtualisation allows common users to get administrative permissions to configure the operating system, networking, and to install and uninstall software packages.
2. The resources (for example, CPU or Memory) can be dynamically assigned and unassigned to the VM at runtime.
3. VM can easily migrate between physical resources at runtime. The migration is transparent to the user. Migration allows distributing VMs at runtime across the resources to increase server consolidation and save energy costs.

The flexibility of Cloud Computing and its success as a business model involves that users with diverse workload requirements access the cloud. Tasks handled by clouds may be CPU-intensive, I/O intensive, memory-intensive, disk-intensive, or even a combination of them.

The high energy costs of current Data Centres require, in addition to an efficient building and hardware design, from the addition of policies and models to assess the VM allocation and management process to minimize the energy costs also from the software perspective.

Deliverable 2.2 comprehends two main research blocks. The first and most extensive block is about developing generalist power models that describe the power consumption of heterogeneous workloads. The second block comprehends the establishment of basic scheduling and management algorithms that aim to reduce the carbon footprint of a Data Centre by means of increasing the energy

efficiency of the workloads (consolidating them) and increasing the ecological efficiency (scheduling the workloads to coincide with the maximum production of renewable energies).

The rest of the document describes the architecture, usage and integration of the four aforementioned main parts of this deliverable. It is organised as follows: section 2 describes part of the research background and experimentation that has been carried to build up the released power models and management algorithms. Section 3 describes the structure of the publicly released prototype and its usage manual to build up new models. Finally the conclusions and future work thoughts are stated.

2 Related Work

Modern approaches for energy-aware IT management rely on the modelling of the power consumption of the current workloads that run within modern hardware architectures [4]. There is a noticeable difference in power consumption when the tasks dominate different resources (CPU, Memory, Network and Hard Disk) [5]. Chen et al. [6] build a linear power model that represents the behaviour of hardware nodes that run Virtual Machines, under high-performance computing workloads. Their model may not be suitable to perform accurate predictions since it relies on hardware nodes. Several authors [7] [8] build power models to infer power consumption that apply to VMs power metering, by using existing instrumentation in server hardware and hypervisors. However, none of these approaches consider the impact of hardware heterogeneity. The work developed in the framework of the RenewIT project considers that computing performance, idle power and power gradient as a function of computing resources are platform-specific [9]. In addition to the hardware heterogeneity, the heterogeneity of workloads (High-Performance Computing, Data-intensive, Real-Time web workloads, etc.) must be considered.

The energy and power models of both hardware and workloads are the foundation for Cloud resources allocation and management algorithms.

Hypervisor-level resource management methods (VM placement, resizing and migration) may be used to improve energy saving of Cloud Data Centres [10] [11]. Consolidating the maximum number of VMs within the minimum number of physical hosts while turning off the idle hosts would minimize the energy impact while maximizing the energy efficiency of the infrastructures. However, this technique would decrease the performance and the Quality of Service of the deployed services [12]. It is required a trade-off between energy and performance that implies to distribute (as opposite to consolidate) VMs with special performance requirements. Most works about consolidation of VMs focus on performance [13] and processor energy consumption [14] but do not consider the energy that is consumed by VM migration [15] (as a result of the intensive use of memory and network that is required to transfer several Gigabytes of the disk and memory of the Virtual Machine from one host to another).

In addition to the logical overhead caused by the overloading of tasks, the interferences between the different tasks that run in the same pool of resources must be considered [9] [16] [17] [18] [19]. Such interference would cause an overhead that consumes extra power and reduces energy efficiency.

When measuring the energy consumption of tasks within the resources, the thermal impact of the tasks must be considered (because the cooling systems

also consume energy). It is required to calculate the relation between temperature and dissipated power for each node to engage thermal-aware temperatures, such as playing with 'hot' and 'cold' tasks to control temperature [20] or placing 'hot' tasks to 'cold' or best cooled processors [21]. This work is independent from the fact that, by physical reasons, CPU temperature itself influences the CPU consumption (the more temperature the more consumption), even if the same work is done. This influence is considered and analysed in the framework of the project (Deliverable 4.6).

When the objective is not only to increase the energy efficiency but also to maximise the ecological efficiency (reducing emissions and pollution), Data Centres that partially operate with renewable energies may schedule their workloads (if possible) according to the availability of such energies [22] [23]. For example, a Data Centre connected to solar panels schedules batch applications to the hours when there is the highest solar radiation. As a complementary policy, Data Centres equipped with renewable energies that could be activated on demand (e.g. biomass, biogas), the facility could adapt energy supply to the expected workload [22].

When operating with different Data Centres that are geographically distributed, IT management policies could also minimise the energy impact and maximise the ecological efficiency if it considers the spot status of each facility [24] [25] [26]: energy price, availability of energy according their sources, or the status of the workloads and the physical nodes.

Finding the optimum allocation of a set of VMs in a large Data Centre to fulfil the Quality of Service requirements while minimizing the energy and ecological impact may be unsuitable because of the combinatorial complexity of optimizing a function with a large number of variables.

OpenStack Neat [27] use four types of policies: to detect nodes underload to migrate out VMs and suspend the node; to detect node overload to migrate out VMs and avoid penalising the Quality of Service; heuristic algorithm that decides which VMs to migrate in the hosts where overload is detected; VM placement based on *bin packing*. Bonde [28] also uses bin packing and compares different heuristics with the objective to minimize the number of physical nodes that are turned on. His scenario has two main limitations: VMs are deployed individually (not in groups) and it assumes that hosts are homogeneous. Xu et al. [29] proposes fuzzy logic methods combined with genetic algorithms to optimize the VM placement by considering contradictory objectives: minimize free space in servers, minimize power, and minimize temperatures. Their model is limited because it only considers CPU load percentage. This project will expand this related work in many facets: considering other resources in addition to CPU, such as network, memory, and disk; consider performance in terms of application metrics (throughput, execution time, response time…); and considering

heterogeneous resources, in terms of performance and consumption, so the most energy-efficient status will not always be the status with less working nodes.

3 POWER MODELS FOR ENERGY ASSESSMENT OF WORKLOADS

This deliverable releases power models that describe the energy consumption of a set of heterogeneous workloads in function of their intensity, usage and resources, and how they share them with other workloads during the consolidation process.

Work Package 2 of RenewIT project considers three types of workloads. Each one has its own particularities:

- **Web** workloads have real-time requirements: the users of such workload need to get a response to their petitions in few seconds (e.g. APDEX standard [30] sets the threshold for frustrated users in 4 seconds). Some examples of *performance metrics* of web workloads are: requests per second, response time, throughput (KB per second), or concurrent users. There is not a typical resource consumption profile for web workloads. They may use CPU, memory, network or disk in several proportions.
- **HPC** workloads are typically CPU-intensive [31]. They perform a large amount of floating-point operations for scientific calculations (while web workloads usually have a highest rate of integer operations). Because HPC workloads may last for hours, even days, they do not have real-time requirements, and are usually allocated in job queues that may execute them hours or days after they are submitted by the users. Typical *performance metrics* for HPC workloads are number of Jobs, Job sizes, average percentage of used CPU and memory.
- **Data** workloads are usually both memory and disk-intensive, while they can also use a high rate of CPU operations (integer or floating-point) for data analysis [32]. Despite of data workloads may have real-time requirements (e.g. a search query in Google or Bing), the work performed in this project considers data workloads without real-time requirements (e.g. background data analytics for business intelligence applications). Typical *performance metrics* include: bytes read/write, data throughput.

In addition to the heterogeneity of workloads (and thus, the system resources they use), the models must also consider other variables, like the architecture/vendor/model of the hardware that executes the workloads and the middleware that is being used between the software and the hardware (since they may impact the measurements).

This task considers the next variables to build a composite power model for energy assessment of heterogeneous workloads:

- **Hardware Platforms**. Deliverable 2.1 released a micro Data Centre based on Intel Xeon E5-2640 processors from year 2012. However, the model generation process has been tested in four additional architectures: AMD Opteron 6140 and 6234, and Intel Xeon E5-2650 and X5650.
- **Middleware**. The models must quantify the impact of the intermediate software layer in the system measurements. Next middlewares are considered:
    - Native system: the workloads are executed directly in the operating system (Linux) of the hardware nodes.
    - KVM: Kernel-based Virtual Machine hypervisor that is bundled by default in modern Linux kernels. It provides hardware-based full virtualization [33] to allow the execution of unmodified guest VMs.
- **Usage of system-level resources**. Their measurement comprehends a wide range of OS-level metrics and hardware counters that measure the number of operations of the subparts of the system, for example:
    - CPU
        - Number of integer instructions
        - Number of floating-point instructions
        - Number of Single Instruction Multiple Data (SIMD) instructions
        - Number of load/store operations
    - Cache Memory (Levels 1, 2 and 3)
        - Capacity
        - Bandwidth
    - Main Memory
        - Number of accesses
        - Bandwidth
    - Disk
        - Number of accesses
        - Bandwidth
    - Network
        - Packets sent/received

- Bandwidth

Please refer to Annex B for a complete and detailed list of low-level resource counters.

Figure 3.1 shows the iterative workflow for the construction of power models for energy assessment of heterogeneous workloads, according to the aforementioned variables and measurements:

1. Execute micro-benchmarks, like iBench [34], which allow to selectively stressing individual resources of the system (for example, stress only the floating point unit of the CPU, or stress only the cache memory). This set of micro-benchmarks minimizes the interferences between system resources and allows creating simple power models of the individual system-level resources.
2. To select a set of representative system-level resources to create a composite, generalist model, that will be able to predict the consumption of a workload given the usage patterns of the system-level resources. To avoid providing a too complex model that becomes unmanageable, the system-level resources that impact minimally in the general power consumption will be discarded and not included in the final model. Please refer to Annex B for a complete list of the system-level resources that have been considered in our models.
3. Characterise workloads (Web, HPC, Data) to extract patterns of their typical usage of system-level resources. Benchmarks that mimic real Cloud applications (e.g. CloudSuite [35]) have been used to extract resource usage profiles.
4. Validate the composite model by executing again the workloads. The measured power consumption is compared with the power consumption of the model. Section 3.2 describes the first validation results.

This process is repeated iteratively. The output of each task is used as input for the next task to (1) refine power models of the individual system-level resources; (2) refine the characterisation of the different workloads; (3) refine the composite model by quantifying the relevance of its parameters, as well as their interference/relation.

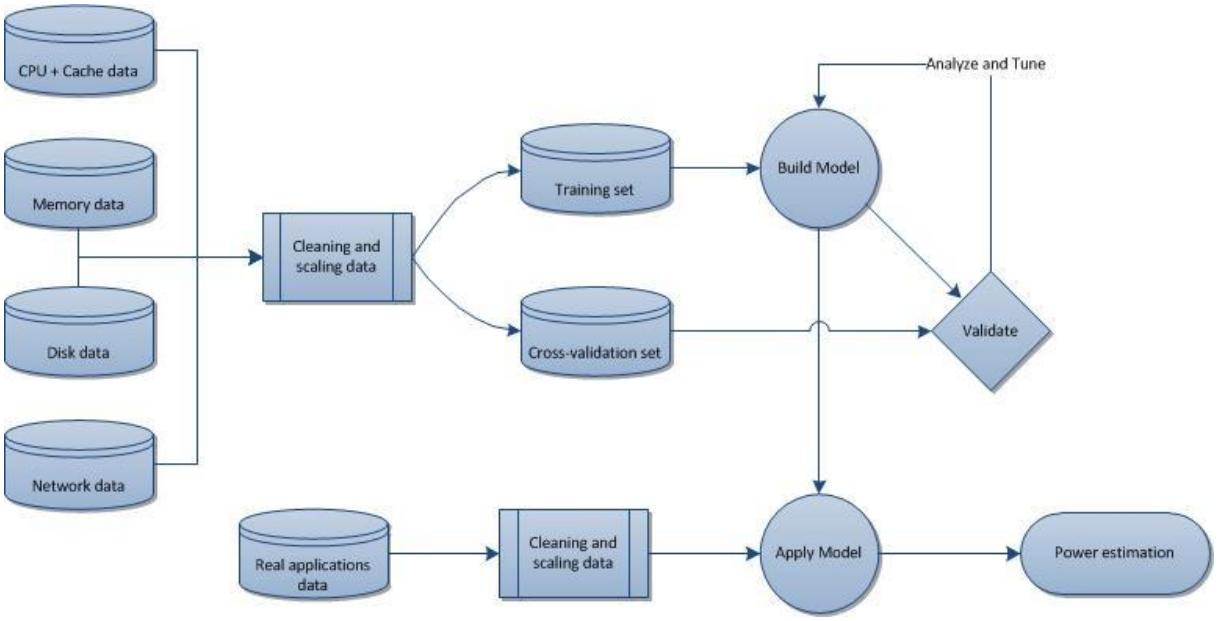


Figure 3.1: Workflow for power models generation

3.1 Experimental Background

The following experiments show an initial quantification of the individual impact of system resources (CPU, memory, disk, network) on the overall system power consumption. By themselves, they are not useful to predict the power consumption, because real workloads use a combination of system resources. However, they provide data to train the overall power model.

The graphs of this section show the power profiles for the Intel(R) Xeon(R) CPU E5-2650 at 2.00GHz. The graphs for the other architectures and models are omitted to avoid redundancy, because they follow similar shapes (only absolute values are different).

CPU and Cache Memory

The next micro-benchmarks have been executed:

- Ibench suite [34]: 3 benchmarks from this suite have been adapted in order to stress different types of instructions and perform different type of operations (floating-point multiplications and divisions, integer and square root operations).
- Stress-ng [36]: start a variable number of processes spinning on square roots of random numbers and cache memory stressing.
- Sysbench [37]: calculation of prime numbers using 64-bit integers.
- Prime95 [38]: finding Mersenne prime numbers.

All these experiments run for some hours using different number of CPU cores and sockets in order to stress the CPU at different intensities, covering as many cases as possible and providing the possibility to analyse the impact over the power of each type of operations executed.

Figure 3.2 shows how the power consumption of a single CPU evolves as the load increases. It shows similar power curves for different operations: integer (int), floating point (fp) and square roots (sqrt). However, at high intensity, integer operations slightly consume less power.

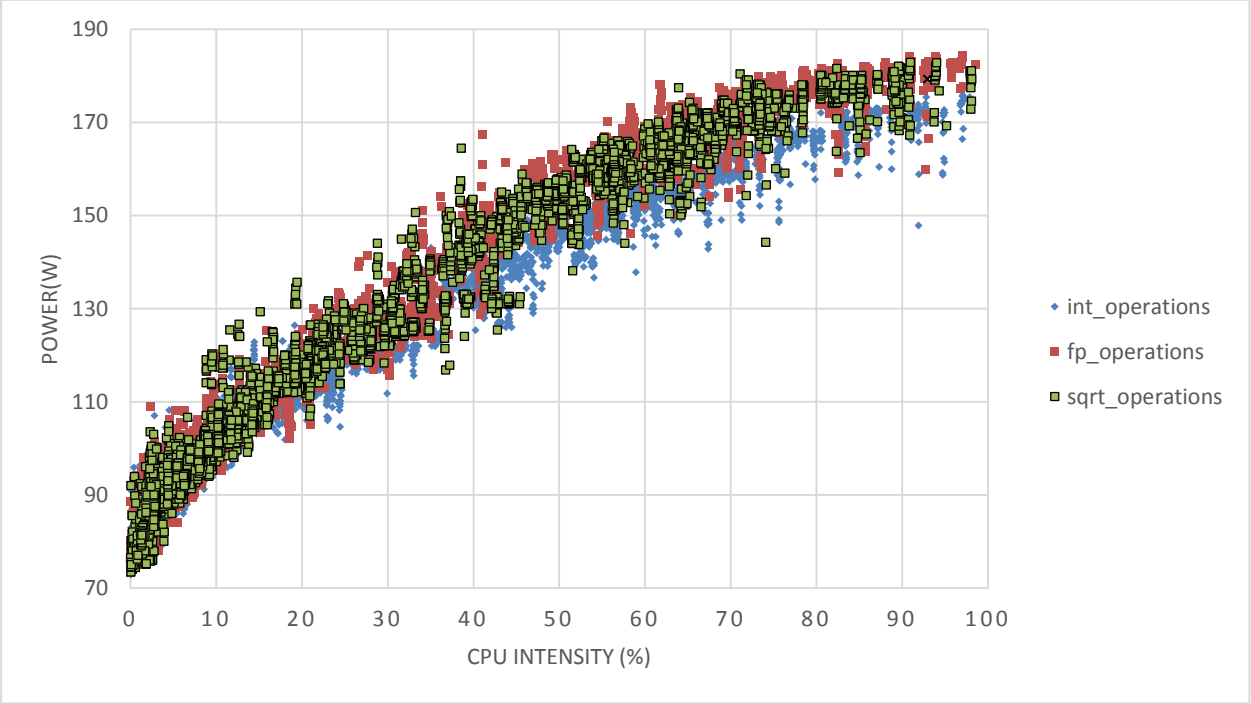


Figure 3.2: Power versus CPU load for different types of operations

For the same workload, the CPU power consumption varies depending on the number of cores and threads that are enabled. For example, Figure 3.3 shows that a process that stresses a single physical core at 70% (1 CPU – 1 THREAD) consumes more power than two processes running in the same core at 35% exploiting the simultaneous multi-threading technology (1 CPU – 2 THREAD).

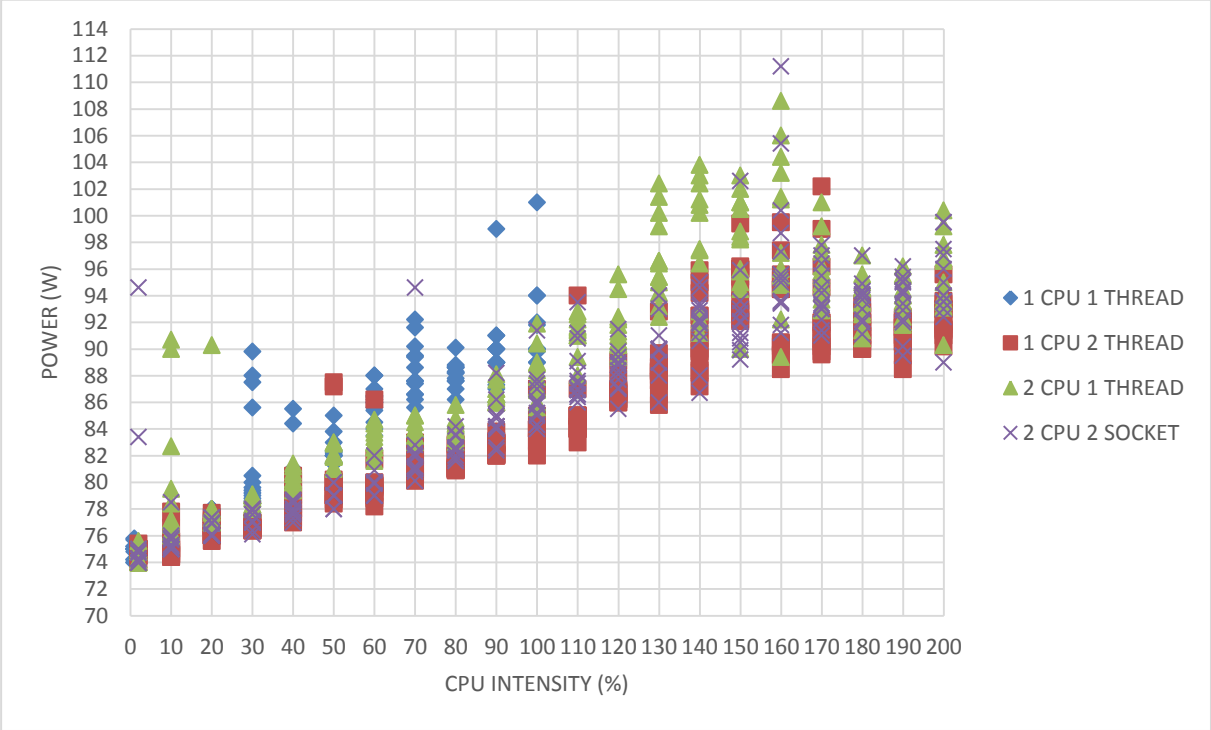


Figure 3.3: Power versus CPU intensity with different number of threads and CPUs enabled. 200% of CPU intensity means 2 cores at 100%

The conclusion of the experiments is that there is a strong influence of the cache in the power. However, Figure 3.4 shows that the proportion between cache loads and power depends on the type of benchmark that is executed. Other resources can also influence the power, since the same number of accesses has different power measurements for different benchmarks.

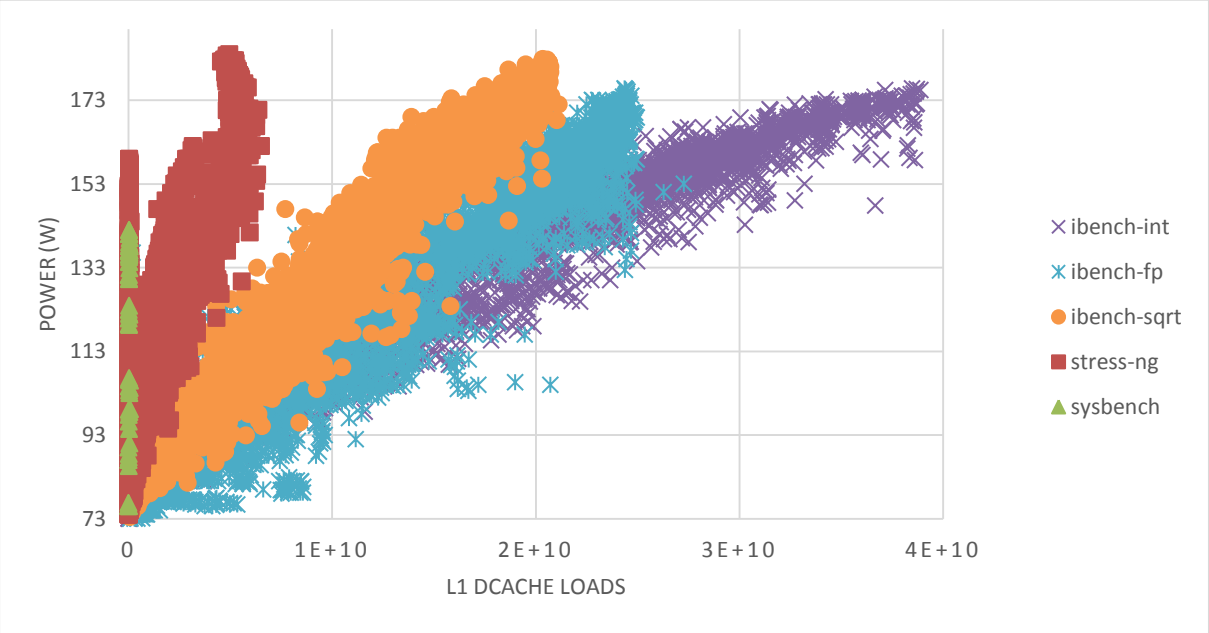


Figure 3.4: Power versus number of Level-1 cache loads for different benchmarks

Main Memory

Different types of experiments have been performed to analyse the relation between power and RAM usage at different levels of intensity, by using the following benchmarks:

- Ibench memory capacity benchmark.
- Stress-ng memory: start a variable number of workers spinning on mapping files on memory (anonymous *mmap* command).
- Pmbw [39]: set of assembler routines to measure the parallel memory.

These experiments stress both the memory capacity and bandwidth (rate at which data can be read from or stored into a semiconductor memory by a processor). Each benchmark runs from 1 to 3 hours, changing incrementally its own intensity from 0% to 100%.

Figure 3.5 shows that power remains constant whatever the used memory is. However, Figure 3.6 shows that memory bandwidth has an almost-linear influence on power consumption.

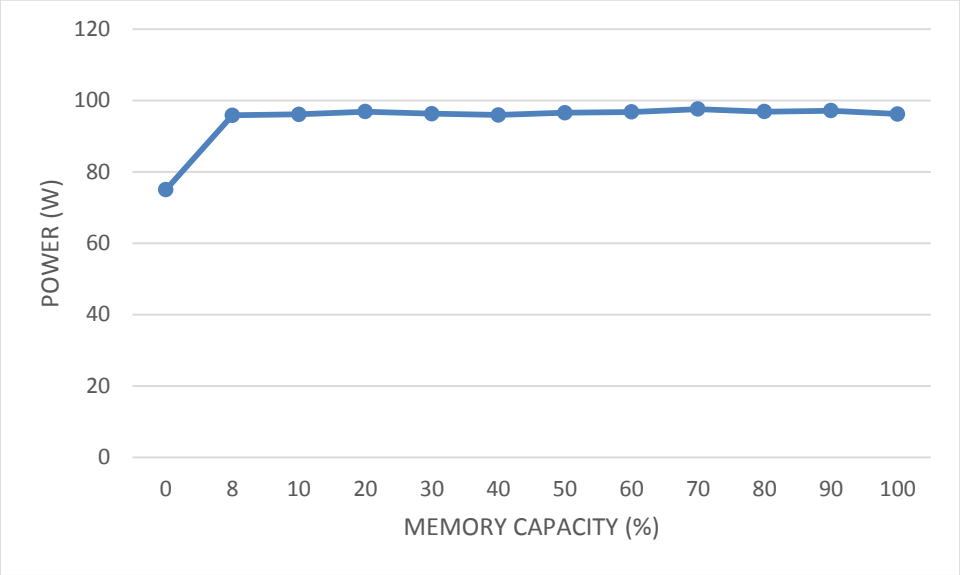


Figure 3.5: Evolution of Power versus the used memory

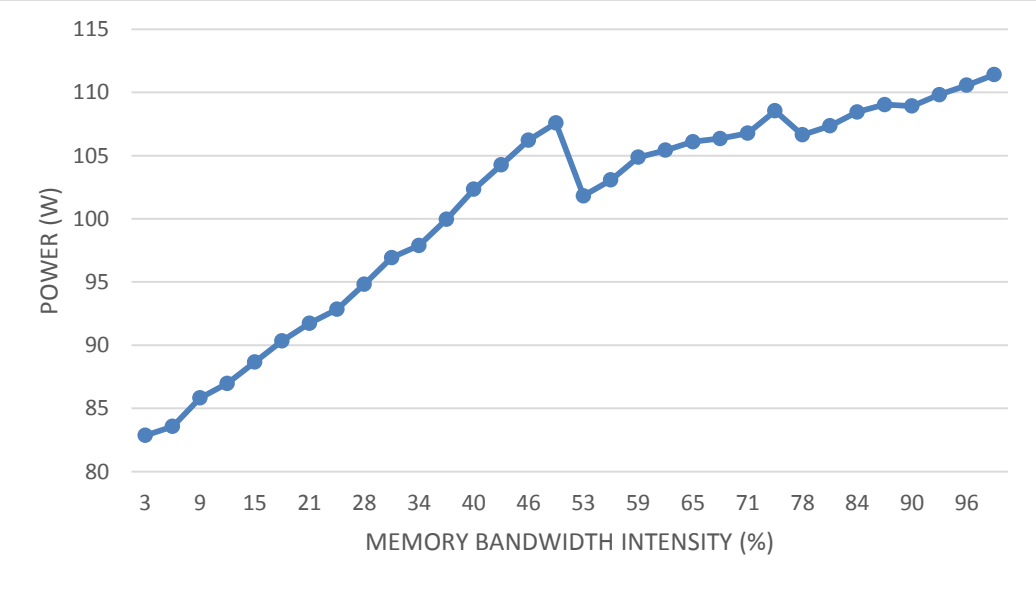


Figure 3.6: Evolution of Power versus the bandwidth of the memory

Disk

As for the memory subsystem, rotational disk capacity and bandwidth have been both tested by performing four different types of operation (read, write, random-read and random-write) for the next benchmarking tools:

- Ibench disk capacity benchmark.
- Stress-ng disk: starting a variable number of workers spinning on write operations followed by unlinking the file (this is, detaching the entry from the file system).

- Fio [40]: to spawn a number of threads or processes doing a particular type of I/O actions.

For rotational disks, Figure 3.7 and Figure 3.8 show that both capacity and bandwidth have almost no influence in the global system consumption.

Regarding to disk bandwidth, only the step from 0 to >0 intensity has some influence, because it makes the rotational disk to activate its mechanical parts, which remain consuming the same power during the whole disk operation.

The bandwidth for Solid-State Disks will influence the power similarly as main memory (Figure 3.6).

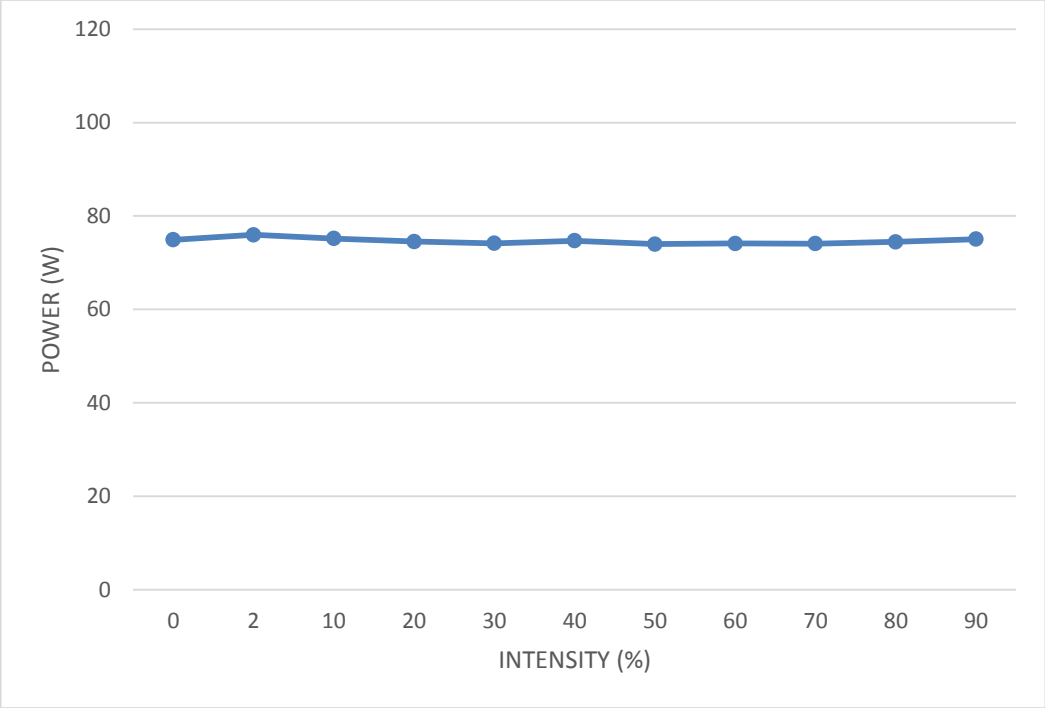


Figure 3.7: Influence of disk capacity over the power consumption

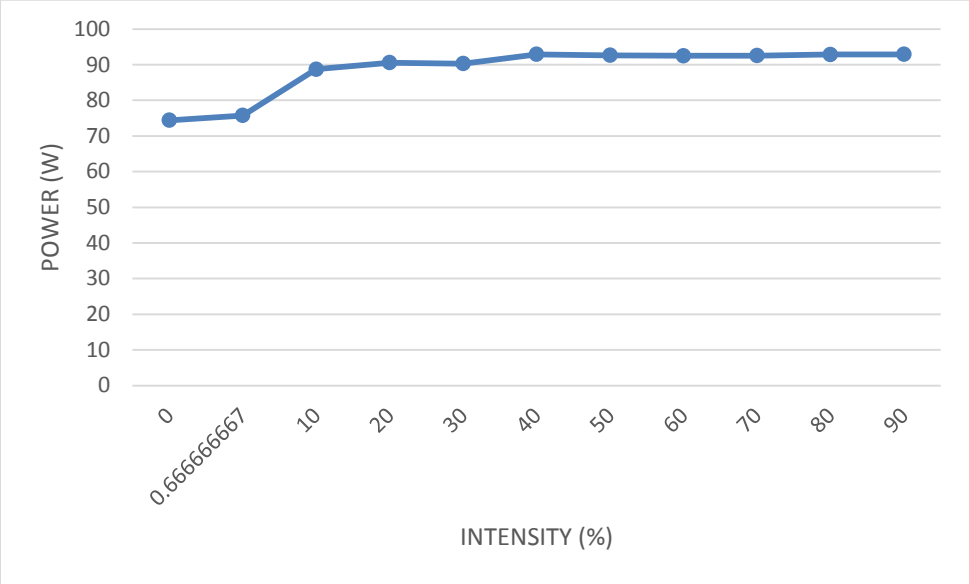


Figure 3.8: Influence of disk bandwidth over the power consumption

Network

Network influence over the power has been tested using the Iperf3 [41] benchmark: an increasing amount of data is sent or received by the server while the energy consumption is measured.

Figure 3.9 shows that network traffic intensity affects almost linearly the power, both for read and writing operations.

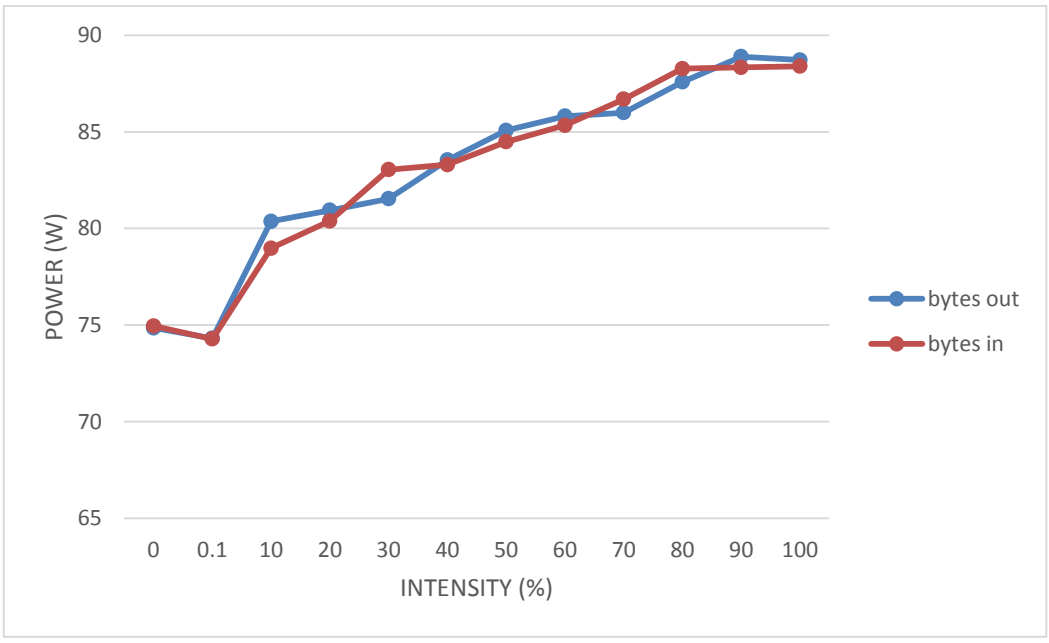


Figure 3.9: Influence of network bandwidth over the power consumption

3.2 Model generation and Validation

Starting from all the data collected using a monitoring system able to capture both high-level system metrics (CPU usage, packets sent/received,...) and low-level metrics from hardware counters (number of FP instructions executed, accesses to LLCache...), different machine learning and pattern recognition techniques have been used to quantify the contribution of each subsystem metric to the energy consumption of a server (Power). The initial set of experiments are analysed by means of the Weka data mining software [42].

The model is evaluated and built experimentally and, using an iterative process to tune up the parameters and choose different metrics and algorithms, the error of predictions is decremented (Figure 3.10). In addition to the error of prediction, the model must also consider the execution time that is needed to elaborate the data, to allow its application into a real-time environment.

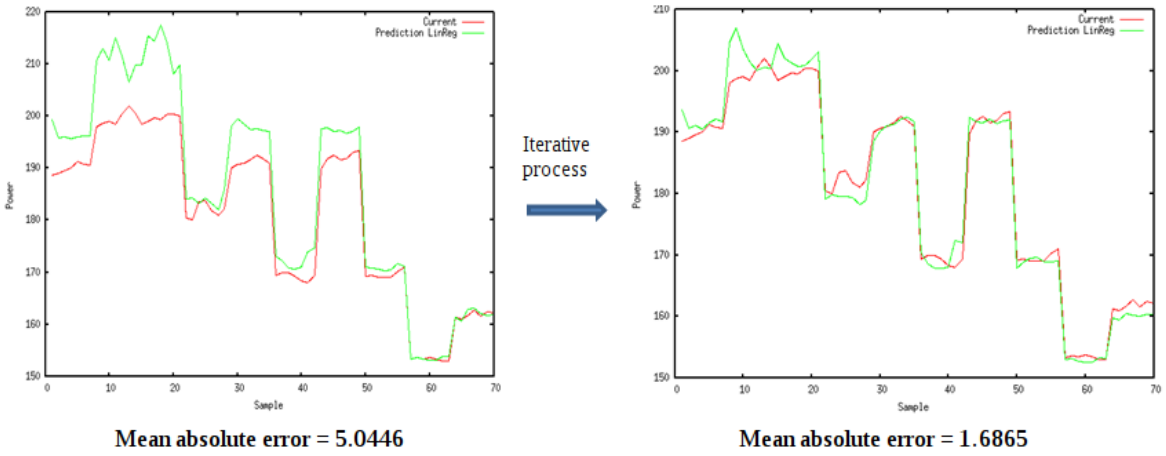


Figure 3.10: Reduction of the error through an iterative process

After evaluating many Machine Learning algorithms to build up the model, different accuracies have been observed, depending on the used training set of data:

- **REPtree**: Fast decision tree learner. Builds a decision/regression tree using information gain/variance and prunes it using reduced-error pruning (with back-fitting). Only sorts values for numeric attributes once. Missing values are dealt with by splitting the corresponding instances into pieces.
- **Bagging**: an ensemble method that creates separate samples of the training dataset and creates a classifier for each sample. The results of these multiple classifiers are then combined (such as averaged or majority voting).

- **Multiple Linear Regression**: it attempts to model the relationship between two or more explanatory variables and a response variable by fitting a linear equation to observed data. Every value of the independent variable *x* is associated with a value of the dependent variable *y*.
- **Neural Networks**: a model with 4 hidden layers and 20 neurons is self-trained by comparing the output of the neural network with its predicted values, then the weights of the connections between neurons are adjusted.

Both Reptree and Bagging are accurate when the training set contains all the possible cases because it is not able to predict values that are not in the training set. Multiple Linear Regressions do not work accurately if the correlations between variables are not linear. The model has finally been built with Neural Networks, since they can capture nonlinear, hidden behaviour of the resources.

Model validation

Once the micro benchmarks have been used to train the model, such model is validated with benchmarks that reproduce real applications. Two benchmarks have been used:

- Cloudsuite [35] is a benchmark suite for emerging scale-out applications. The second release consists of eight applications that have been selected based on their popularity in today's Data Centres. The benchmarks are based on real-world software stacks and represent real-world setups: data analytics, data caching, data serving, graph analytics, media streaming, software testing, web search and web serving.
- NAS Parallel Benchmarks [43] are a small set of programs designed to help evaluate the performance of parallel supercomputers. The benchmarks are derived from HPC applications, in particular from computational fluid dynamics applications. Task 2.3 used three different implementations of the benchmarks: sequential execution mode, and two parallel programming models (MPI and OpenMP).

Figure 3.11 and Figure 3.12 show the initial validation of the Neural Network model with 4 hidden layers, for different datasets of the Intel Xeon and AMD Opteron processor families, respectively. Each graph shows the relation between the predicted and measured power. The "training" graph shows the validation of the model, which has been trained with the micro benchmarks, when it predicts the consumption of the same micro benchmarks (it is validated with the training set as input). The "Validation" graph shows the prediction of the power consumption of the same model when predicting the consumption of the Cloudsuite and NAS benchmarks (it is validated with different input than during the training phase).

It can be observed that real-application benchmarks are not able to stress the resources as much as micro-benchmarks, so real-application benchmarks never reached power measurements as high as micro benchmarks. The explanation for that is that real application benchmarks have to coordinate the load of various low-level resources. CPU, which is often the resource with the highest consumption, must often wait for network, disk and memory operations.

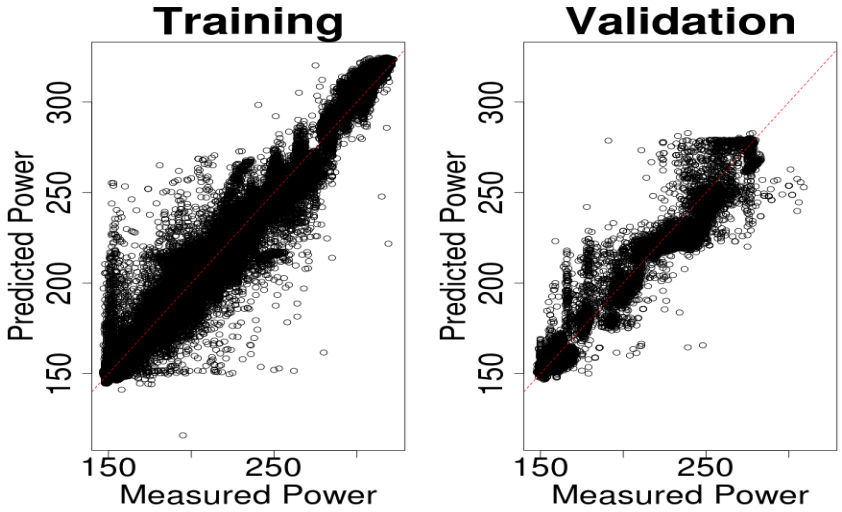


Figure 3.11: Predicted vs Measured power in Intel X5650 with training and validation datasets. Correlation = 0.983, Mean Average Percent Error = 2.44%

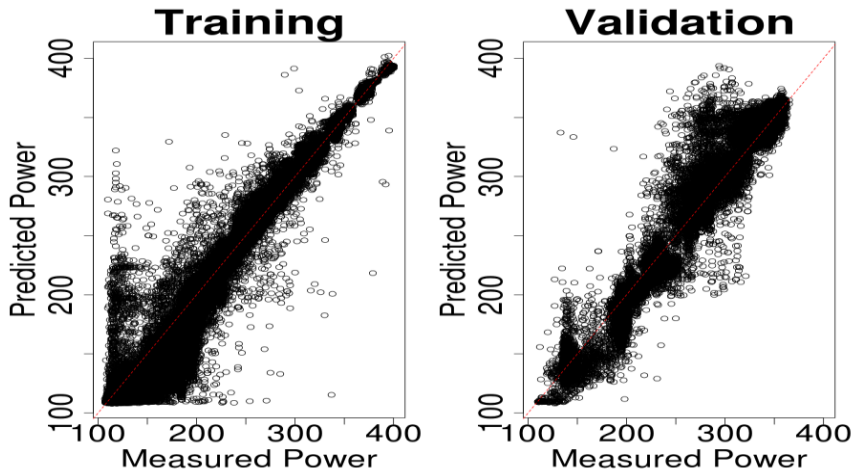


Figure 3.12: Predicted vs Measured power in AMD Opteron 6140 with training and validation datasets. Correlation = 0.984, Mean Average Percent Error = 3.68%

Figure 3.13 shows the actual power measurements and the predicted values during the time, while different benchmarks are executed. The first part of the graph (until sample ~7500) corresponds to the execution of the different

Cloudsuite benchmarks. After that, until sample ~26500, it executes the NAS parallel benchmarks in sequential mode (1 CPU). The other two differentiated blocks correspond to OpenMP and MPI executions of NAS, in that order.

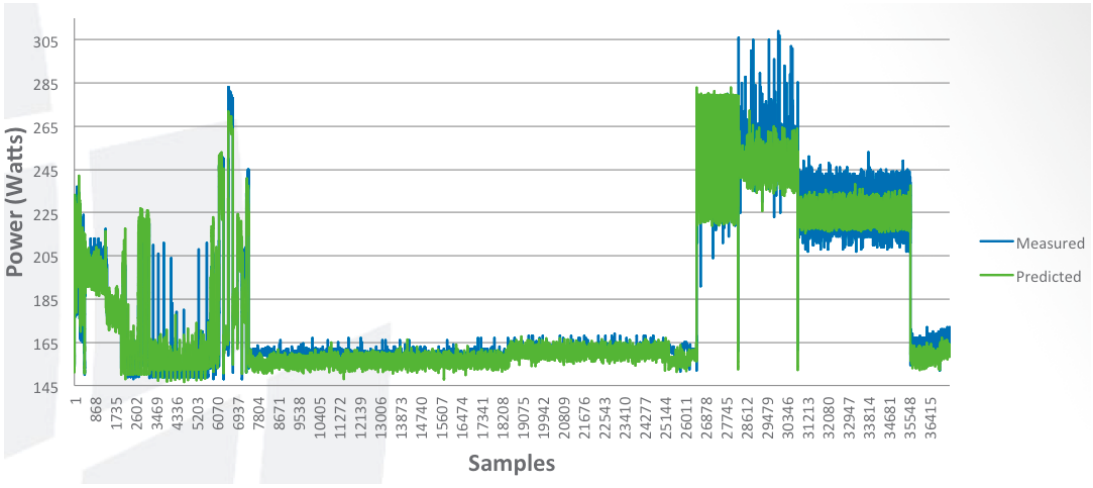


Figure 3.13: Power Model estimation: Measured VS Predicted

Please refer to Annex A for a statistical description (correlation and mean absolute percentage error) of the precision achieved by the power models in comparison with the measured power values for the different hardware platforms that have been tested.

3.3 Experiments with Virtual Machines

Following the same approach described for the physical host, the developed power modeller is also able to estimate power consumption of virtualized environments.

The same set of benchmarks in a VM with 16 virtual CPUs, 16 GB of RAM and 10 GB of Disk were executed.

As for the physical host, Neural Networks were used to build the model and, in order to choose the parameters, such as polynomial degree of its inputs and number of neurons, learning curves were used.

Figure 3.14 represents a plot of the training, cross-validation and test error as a function of the number of neurons used in the neural network. This plot can give a quantitative view into how beneficial it will be to add or reduce the number of neurons: too many neurons will lead to overfit the training set, while few of them will generate a bigger error.

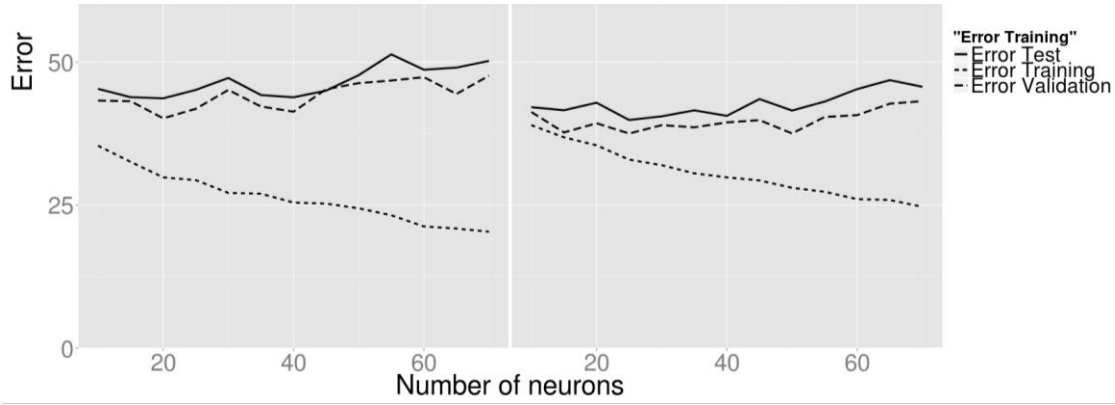


Figure 3.14: Training, cross-validation and test set error as a function of the number of neurons used in the neural network.

Principal Component Analysis (PCA) is also being applied in order to reduce the number of inputs of the model. This algorithm linearly transforms the data in such a way that the components are sorted depending on the amount of information (variance) they provide.

As for the physical host, the model was validated with real-world application from Cloudsuite and NAS inside a VM (Figure 3.15 and Figure 3.16). Although the modelling error for VMs is slightly higher than the error for physical hosts, it is planned to adjust the models accuracy during their application in Tasks 2.4 and 2.5 of RenewIT project, when they are integrated with the energy-aware policies for optimizing VM placement and management.

Analogously to the physical host model, Figure 3.17 shows the accuracy of the model when cross-validating them with the training data set.

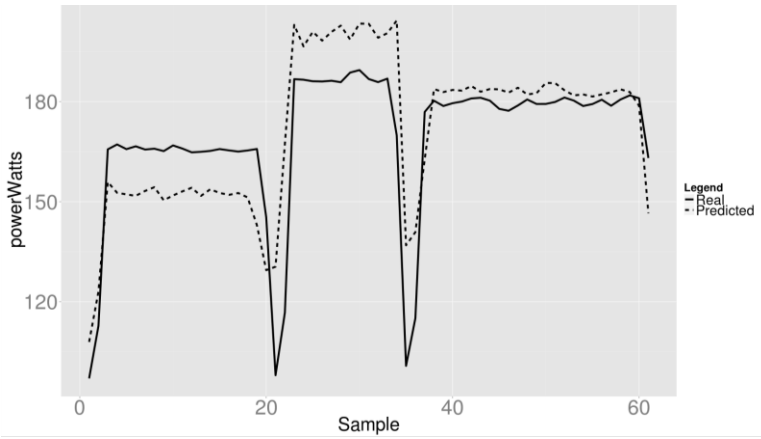


Figure 3.15: VM model validation. Correlation = 0.631, MAPE = 7.69%

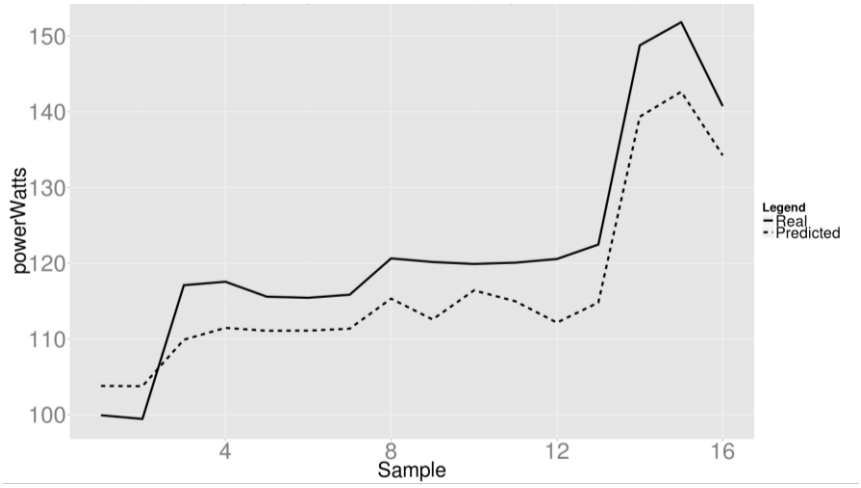


Figure 3.16: VM model validation. Correlation = 0.952, MAPE = 4.95%

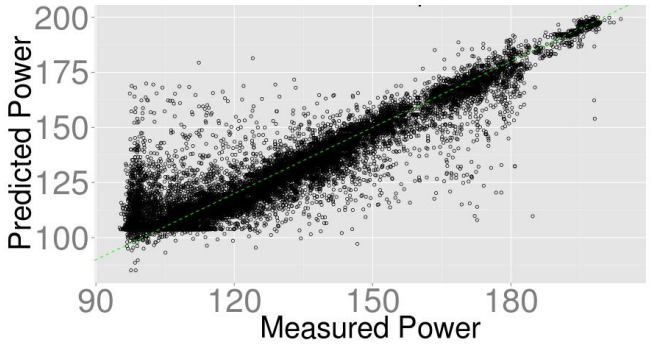


Figure 3.17: Cross validation of Virtual Machine power models. R2=0.881, MAPE=4.33%

3.4 Differences between VM and Host power Models

The later experiments of this task demonstrated that tasks that are executed in VMs and physical hosts show similar power consumption patterns (Figure 3.18 and Figure 3.19).

The three figures show the execution of the NAS Benchmarks (serial and MPI with 9 threads) in a physical host (32-core Intel Xeon E5-2640 v2 @ 2.00GHz) and a 32-CPU virtual machine under the same architecture.

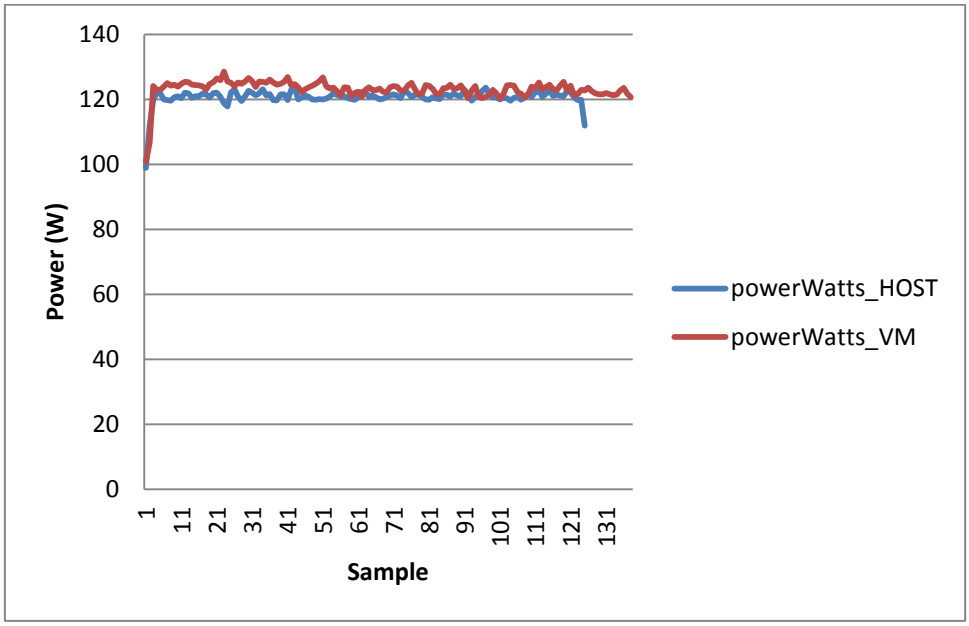


Figure 3.18: NAS Serial power profiles for hosts and VMs

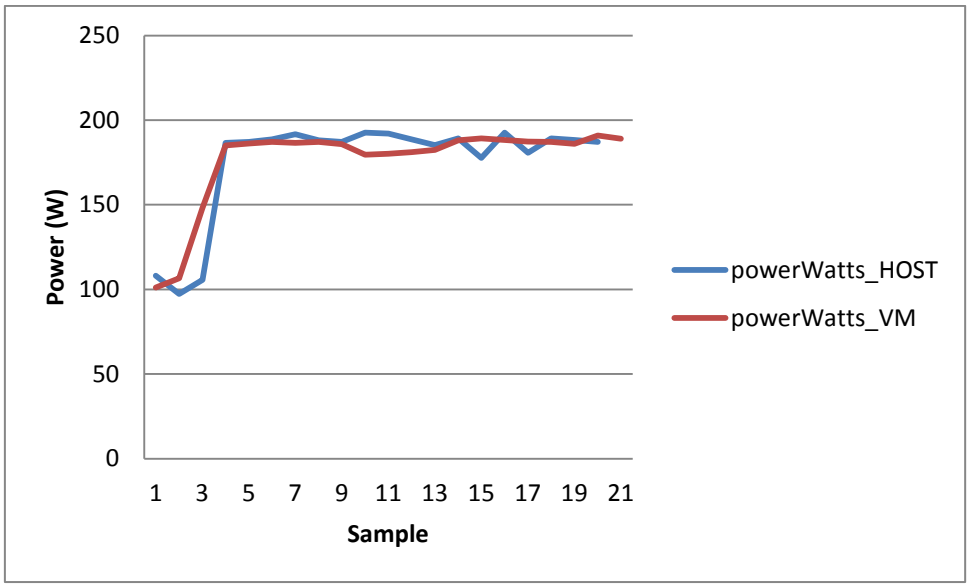


Figure 3.19: NAS MPI (with 9 threads) power profiles for hosts and VMs

Figure 3.18 and Figure 3.19 show that, despite the power profile is almost identical, the performance is lower for the VM use case, as shown by the slightly longer execution time of the experiments for the VM.

Table 1 shows the differences in terms of execution time and application performance (expressed as millions of operations per second) for the serial and parallel benchmarks, for both VM and physical hosts:

Table 1: execution time and operations/second for NAS Serial and NAS MPI benchmarks with both physical host and Virtual Machine

	Physical host	Virtual Machine
NAS Serial		
Execution time (seconds)	1215	1358
Mops/second	2359	2110
NAS MPI (9 threads)		
Execution time (seconds)	174	194
Mops/second	16457	14784

4 ENERGY-AWARE POLICIES FOR OPTIMISING VIRTUAL MACHINE ALLOCATION AND OPERATION

The power models that have been developed in this Deliverable have two main purposes:

1. To contribute to the energy modelling of a complete Data Centre, as the main goal of the RenewIT project.
2. To assess resource management policies that would optimize the allocation and operation of VM within Cloud Data Centres, in terms of performance and energy efficiency.

Task 2.3 started the research on optimization algorithms during the allocation and operation of VMs, which will be completely developed on tasks 2.4 and 2.5.

The first release of the VM Management algorithms has been released as a Java library called Clopla (see section 5.4 for details about obtaining and configuring it), which is based in the OptaPlanner [44] constraint satisfaction solver library.

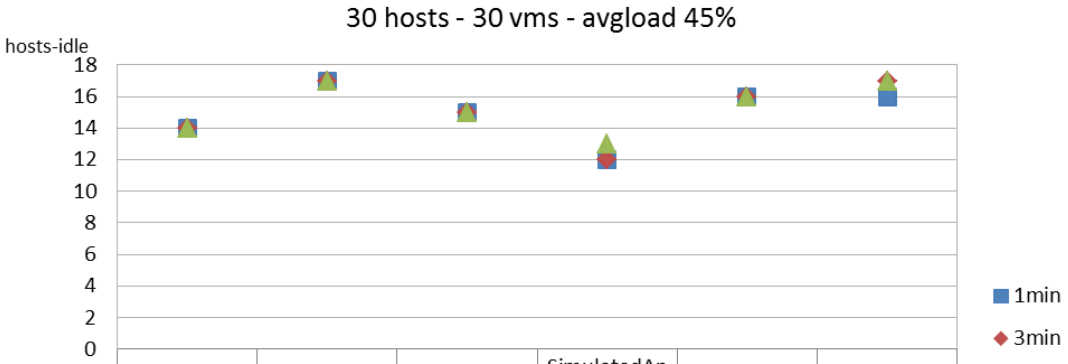
The optimization process can be triggered at three stages:

- A new VM (or group of VMs) must be deployed within a set of hosts. It must be allocated to allow the VMs access all their requested resources (enough free memory, CPUs, disk…) while maximizing the energy efficiency (e.g. consolidating as many as VMs within the same physical host)
- A VM (or group of VMs) must be undeployed. It may happen that the system gets unbalanced (for example, the remaining VMs could be consolidated to increase the energy efficiency rate) and the VMs need to be migrated to optimize the system.
- Because of the associated error during the estimation of the required resources of the VMs, it may happen that the system is unbalanced at a given stage of the execution of the VMs. The VM Management algorithms can be executed periodically to rebalance the system.

Clopla basically configures OptaPlanner to run an optimization search to optimize the placement of VMs, given a set of constraints (e.g. a VM MUST be pinned to a given host, or the specification of the resource limits).

In the framework of Task 2.3 an initial investigation of the promising potential of Clopla and Optaplanner was done. The initial experiments have been focused on

calibrating the best configuration for local search algorithms. Figure 4.1 shows the performance for different local search algorithms for the consolidation of 30 VMs that represent the 45% of the load of 30 physical hosts. The Y axis represents the number of hosts that will remain idle (they can be switched off to save energy) after a VM redistribution (the higher the better). The results concluded that "late acceptance" search is almost always the best local search algorithm. The algorithms have been tested with three different execution times to search for a solution: 1, 3 and 5 minutes. Theoretically, the more time the system have to look for a solution the better solution it will find, but in practice any remarkable difference between 1 minute and 5 minutes was observed.



	HillClimbing	LateAcceptance-400	LateSimulatedAnnealing-100-1000	SimulatedAnnealing-2hard/100soft	StepCountingHC-400	TabuSearch-7-1000
1min	14	17	15	12	16	16
3min	14	17	15	12	16	17
5min	14	17	15	13	16	17

Figure 4.1: Comparison on the effectiveness of different local search algorithms, for different constrained search times, for VM consolidation

Since the explanation of the concepts behind the local search (hill climbing, late acceptance, late simulate annealing, tabu search…) are long and completely out of the scope of this deliverable, please refer to OptaPlanner construction heuristics and local search configuration document [45] for more details about the optimization details.

5 STRUCTURE OF THE PROTOTYPE

5.1 INTRODUCTION

Deliverable 2.2 is, according to the grant agreement of RenewIT project, a public prototype. Therefore, it has been publicly delivered in the next repository:

https://github.com/bsc-renewit/d2.2

The main purposes of this section are:

1. To describe the structure and architecture of the public prototype that has been released to the community in a public source code repository:
2. To provide a theoretical background to the experiments that have been performed to build up the power models that are released D2.2.
3. To release a user manual that would allow project-external people to rebuild the power models, and building their own custom models, adapted to the particularities of the hosts they own.
4. To describe the work in terms of initial VM Management optimisation, which will be continued in D2.3, and provide some guidelines that would allow IaaS providers to integrate our algorithms within their cloud managers.

Following schema describes the directories structure of the prototype repository:

```
D2.2
├── algorithms
│   └── clopla
├── powerModelling
│   ├── dataSets
│   │   ├── AMD Opteron 6140
│   │   ├── AMD Opteron 6234
│   │   ├── INTEL Xeon E5-2640 RenewIT
│   │   ├── INTEL Xeon E5-2650
│   │   └── INTEL Xeon X5650
│   └── prototype
└── monitoringFramework
```

- **monitoringFramework** contains a metric collection framework developed at Barcelona Supercomputing Centre (BSC) to gather power and performance metrics, in addition to the metrics that are already provided by the Ganglia and sFlow monitoring agents.
- **powerModeling/datasets** contains the data that has been collected during the execution of benchmarks in different host environments (two

different models of AMD Opteron and three different models of Intel Xeon processors). Please refer to Annex B for a detailed list of the measured data.

- **powerModeling/prototype** contains the prototype implementation of the power model generator that allows to estimate and predict the power consumption of a server.
- **algorithms/clopla** contains the Clopla java library that, given a set of virtual machines and hosts, computes the optimized placement for the VMs.

5.2 MONITORING FRAMEWORK USER MANUAL

The monitoring-framework is a metric collection framework developed at BSC.

It gathers metrics additionally to the ones collected by Ganglia Monitoring Framework daemon [46] or Host sFlow [47], and injects them to Ganglia. It collects hardware performance events counters (host and VM level), raw performance events counters (host and VM level), usage per core and power metrics from several sources and sensors. It can also collect additional metrics that depend on the sensor availability in each server, like temperature metrics.

The performance and raw performance event counters are collected by means of Perf, a profiler tool supported by Linux 2.6+ based systems. The Perf tool can measure events coming from different sources such as pure kernel counters (context-switches, minor-fault, etc.) and micro-architectural events (number of CPU cycles, instructions retired, level 1 cache misses, last level cache misses, etc.). The raw performance event counters are additional CPU counters that Perf does not list out-of-the-box as named counters. Examples of them are the number of integer or floating point operations executed. To capture them, its hexadecimal code needs to be find out using the tool *perfmon2/libpfm* (described later) and supplied to perf to capture them.

5.2.1 REQUIREMENTS

The next Linux packages must be installed (for Debian-like distributions):

- linux-tools-common
- lm-sensors
- python-dev
- flex
- bison
- wget

- python-pip
- python-guestfs
- python-pexpect

The next python packages are required (installed with the *pip install* command):

- pexpect
- pysnmp
- psutil

To activate the counter monitoring with Perf tool, first check current perf version supports the command 'perf kvm stat' (just type it in a terminal and check the output). If it does not, download and install a newer version and set the path in the configuration file. Type the next commands in the Linux terminal:

```
sudo apt-get install flex bison
wget   http://ftp.de.debian.org/debian/pool/main/l/linux-tools/linux-tools_3.13.6.orig.tar.xz
tar xvf linux-tools_3.13.6.orig.tar.xz
cd linux-tools-3.13.6/tools/perf/
make
```

Set perf_tool in the configuration file

Finally, to get the temperature metrics (lm-sensors) accessible from Python, the PySensors is required. Just type the next commands from the linux terminal:

```
wget   https://pypi.python.org/packages/source/P/PySensors/PySensors-0.0.2.tar.gz
tar xvf PySensors-0.0.2.tar.gz
cd PySensors-0.0.2/
sudo python setup.py install
```

To install Perf tool, get the latest version of perfmon2/libpfm:

```
git clone git://perfmon2.git.sourceforge.net/gitroot/perfmon2/libpfm4
cd libpfm4
make
```

5.2.2 CONFIGURATION

The file `extraMetrics.conf` is the configuration file for the monitoring system:

The next properties accept 'yes' or 'no' as values to monitor the following subsystems: `vm_metrics`, `vm_counters`, `vm_raw_counters`, `host_counters`, `host_raw_counters`, `temperature_metrics`, `power_metrics`.

To monitor `vm_metrics` (VM memory, cache and swap usage) it is required to set `vm_file_path` to the path of the file within the VM that stores those information.

Set `metrics_config_path` property to the file containing the metrics configuration (metrics.conf).

To configure the Perf raw counters:

1. Run the `showevtinfo` program (in examples subdirectory of the downloaded perfmon2 tool sources) to get a list of all available events, and the masks and modifiers that are supported (see the output below for an example of the full output)
2. Figure out what events and what with masks and modifiers you want to use. The masks are prefixed by Umask and are given as hexadecimal numbers and also symbolic names in the square brackets. The modifiers are prefixed by Modif and their names are also in square brackets.
3. Use the check_events program (also in examples sub-directory) to convert the event, umask and modifiers into a raw code. You can do this by running the command as:

   ```
   check_events <event name>:<umask>[(:modifers)*]
   ```

   i.e., you supply the event name, the umask and multiple modifiers all separated by the colon character. The program will then print out, amongst other things, a raw event specification, for example:

   ```
   Codes          : 0x531003
   ```

   This hexadecimal code can be used as parameter to GNU/Linux perf tools, for example to perf stat by supplying it with -e r531003 option

5.2.3 Execution

Launch the monitoring scripts that are provided in the monitoringFramework folder of the deliverable:

```
sudo ./start.sh
```

To stop the monitoring scripts:

```
sudo ./stop.sh
```

The log messages are located in the file:

```
extraMetrics.log
```

5.3 Power modelling User Manual

This tool contains a power model generation for energy assessment of heterogeneous workloads that allows estimating and predicting power consumption of a server. Different types of model have been implemented and can be used in order to develop a generic power model:

- Resources model: following this approach, the power consumption is computed as the sum of the power consumed by the main subsystems of a host:

$$P_{tot} = P_{cpu} + P_{memory} + P_{disk} + P_{network}$$

- Global model: This approach does not consider each resource (CPU, memory, disk and network) separately but the relationship between two or more explanatory variables using all the data acquired during the micro-benchmarks execution as a training set.
- Combined model: The combined model is made up of 2 steps:
    1. Generate a model for each subsystem.
    2. Combined the generated models using the same or a different algorithm.

5.3.1 CONFIGURATION

Configuration files, if not present, will be automatically created the first time the tool is executed. If the environment variable PMG_HOME is not set, the default directory /opt/PowerModelGenerator will be used.

configuration.properties:

```
csv-delimiter=,
independent=powerWatts

# If new-model is set to true, a new model is computed. Set the path
# of the model otherwise.
new-model=true

# If true data will be scaled and standardized
scale-data=false

# If new-model is set to false, set the path of the model to be used
model-path=/temp-1428915018899/serializedGlobalModel.model
# Model types available: resources, global, combined
model-type=global

# If resources model is used, up to what level do you want to build
# the model
# Possible options: cpu, cpu+mem, cpu+mem+disk, cpu+mem+disk+net
# Note that the resulting model will be validated with the dataset
# specified in the     #"validation"
# property of datasets.properties.

model-resources-level=cpu+mem+disk

################Model Options  ################
```

```
# Classifiers available: linearregression, reptree, multilayerperceptron,
# bagging
classifier=multilayerperceptron

# If combined model-type is chosen, set the algorithm to be used in the
# 2nd  step.
# Classifiers available: linearregression, reptree, multilayerperceptron,
# bagging
step2-classifier=linearregression

# If preprocess-dataset is true, set the filter-type to be used
preprocess-dataset=false
#### Filter types available: movingaverage,removeidle ####
filter-type=movingaverage

###############Filter Options ################
# if removeidle is chosen, set the idle power for training and
# validation set
power-idle-training=75.0
power-idle-validation=75.0
# if movingaverage is chosen, set the window size
moving-average-window=5
```

datasets.properties

Set the files path to be used in order to build the model and to validate it. If model-type has been configured as global the "trainingGlobal" variable must be set. Otherwise set the following variables: "trainingCPU", "trainingMemory", "trainingDisk", "trainingNetwork" with the corresponding datasets to be used.

"validation" contains the path of the dataset to be validated.

Variables syntax: ${var1}, ${var2}

NOTE: All the datasets must contain the same variables.

model-*.properties

`model-cpu.properties`, `model-memory.properties`, `model-disk.properties`, `model-network.properties` contain the variables to be used in the model building.

`model-global.properties` will be used if model-type has been configured as global.

All the variables must be contained in the csv.

Syntax: 3 assignments can be done: metrics, var.NAME, newmetric.NAME

- metrics=METRIC1, METRIC2,.... METRIC1, METRIC2... must correspond to variables present in the csv headers. These variables will be used in the

model generation. Please refer to Annex B for a complete list and description of the metrics.

- var.NAME=METRIC1+METRIC2... can be used to build new metrics that can be used in the "newmetric.NAME" assignment. Metrics names existing in the csv must be included within squared brackets: E.g. {cpu_user}. These variables will NOT be used in the model generation unless used in the "newmetric" assignation.
- newmetric.NAME={METRIC1}^{2}+{METRIC2} It is possible to create new combinations of metric that will be used in the model generation. Metrics names existing in the csv must be included within squared brackets: E.g. {cpu_user}. Exponentiation, Squared roots and Logarithmic functions can be used: E.g. log({cpu_user})

Do not modify *.r files. They are used for filtering and error calculation.

5.3.2 Execution

Requirements: Maven, Java 7 In the "prototype" directory run 'mvn package'. Once successfully compiled and configured (see section above), run the tool:

```
java -cp bsc-powermodeller-1.0-SNAPSHOT.jar
es.bsc.autonomic.powermodeller.Main
```

Output: After details about the model generated, several information about errors and correlation will be printed:

- Correlation
  - R^2 - coefficient of determination R^2
  - MAE - Mean Absolute Percentage of mmetric forecasting metric
  - RMSE - root mean squared error
  - RAE - relative absolute error
  - MAPE - Mean Absolute Percentage of mmetric forecasting metric

Serialized model and estimation file paths will be printed. The latter one is a csv containing two columns: "Pactual" (measured values), Ppredicted (preditcted valued).

A graph showing actual vs predicted power is also generated.

5.4 Clopla: Energy-aware policies

Clopla is a Java library that, given a set of virtual machines and a set of hosts, computes an optimized placement for the VMs.

Clopla supports:

- Several construction heuristics: first fit, first fit decreasing, etc.
- Several local search heuristics: simulated annealing, tabu search, hill climbing...
- Several placement policies: consolidate the VMs, distribute the VMs, place the VMs randomly, and group the VMs by service or application.

Clopla uses OptaPlanner [44].

5.4.1 INSTALLATION

Clopla can be integrated with Maven project. Simply add this dependency to the project pom.xml:

```
<dependency>
    <groupId>es.bsc</groupId>
    <artifactId>clopla</artifactId>
    <version>1.0.0</version>
</dependency>
```

5.4.2 USAGE

Clopla is simple to use. It is only required to define a set of VMs, a set of hosts, and some options for the placement engine: the scheduling policy, the maximum running time, a construction heuristic, a local search heuristic, and whether some VMs are required to be deployed in specific hosts.

For example, suppose that we want to find an optimized placement using the following options:

- Scheduling policy: consolidation.
- Timeout: 30 seconds.
- Construction heuristic: first fit decreasing.
- Local Search heuristic: hill climbing.
- Not interested in pinning some VMs to specific hosts.

The java code for finding a placement using those options is:

```
IClopla clopla = new Clopla();
VmPlacementConfig vmPlacementConfig = new VmPlacementConfig.Builder(
     Policy.CONSOLIDATION, // Scheduling policy
     30, // Timeout
     ConstructionHeuristic.FIRST_FIT_DECREASING, // Construction heuristic
     new HillClimbing(), // Local Search heuristic
     false) // Deploy VMs in specific hosts?
```

```
    .build();
System.out.println(clopla.getBestSolution(hosts, vms, vmPlacementConfig));
                                          // get placement and print it
```

The only thing missing from the example is knowing how to instantiate a list of VMs and a list of hosts:

```
// Create a list of VMs that contains a VM with id = 1, cpus = 2,
// ramMb = 1024, and diskGb = 4
List<Vm> vms = new ArrayList<>();
Vm vm = new Vm.Builder((long) 1, 2, 1024, 4).build();
vms.add(vm);  // Instantiate a lists of hosts that contains a host with id = 1,
              // hostname = myHost, cpus = 4, ramMb = 8192,
              // diskGb=100, and that is on
List<Host> hosts = new ArrayList<>();
Host host = new Host((long) 1, "myHost", 4, 8192, 100, false);
hosts.add(host);
```

Every local search heuristic can be configured with different options. It is needed to specify those options when instantiating the local search heuristic. For example, the tabu search heuristic accepts an entity tabu size and an accepted count limit. All these configuration options are very well explained in the Optaplanner documentation [45]

There is a complete usage example in the clopla/src/main/java/es/bsc/clopla/examples folder of the released prototype.

6 CONCLUSIONS

Deliverable 2.2 has released a methodology to build host power models that allow to predict the power consumption of a given host, for different types of workloads (high-performance computing, data-intensive, and web). The models have been validated successfully (mean average percentage error less than 10% in almost all cases) for four processor models from two families (Intel Xeon and AMD Opteron). The measured data, as well as the prototype to implement the models, have been released in a public repository.

This deliverable also releases the initial prototype of energy-aware policies to optimise the placement of virtual machines, which uses the power models in order to minimise the energy consumption of a virtualised Data Centre.

According to the RenewIT roadmap, there is no further work to do with respect to power models. However, prior to its usage within Tasks 2.4 and 2.5, the power modeller is being re-implemented in R language [48], which provides enhanced analytical/statistical tools to improve the accuracy of the predictions.

The work on energy-aware algorithms that has been started in Task 2.3 will be continued in Task 2.4 for optimizing placement of VMs in multiple physical hosts and Task 2.5 for optimizing the selection of placement when there is the choice between multiple Data Centres.

In addition to the definition of new policies and constraints (for example, pinning groups of VMs to a given host, scheduling restrictions for HPC and Data workloads, etc.) future work will also optimize the OptaPlanner configuration to deal with the increased complexity of multiple data centres with many hosts and the application of this number of constraints.

7 REFERENCES

[1] J. Basney and M. Livny, "Deploying a high throughput computing cluster," in *High Performance Cluster Computing: Architectures and Systems, Volume 1*, Prentice Hall PTR, 1999.

[2] R. Buyya, C. Yeo and S. Venugopal, "Market-oriented Cloud Computing: Vision, hype, and reality for delivering it services and computing utilities," in *10th International Conference on High Performance Computing and Communications (HPCC 2008)*, Dalian, China, 2008.

[3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt and A. Warfield, "Xen and the art of virtualization," *ACM SIGOPS Operating Systems Review,* vol. 37, no. 5, pp. 164-177, 2003.

[4] S. Rivoire, P. Ranganathan and C. Kozyakis, "A comparison of high-level full-system power models," in *Conference on Power-aware computing and Systems (HotPower'08)*, San Diego, CA, USA, 2009.

[5] J. W. Smith and I. Sommerville, "Workload classification and Software Energy Measurement for Efficient Scheduling on Private Cloud Platforms," in *1st ACM Symposium on Cloud Computing (SOCC)*, Cascais, Portugal, 2011.

[6] Q. Chen, G. P., K. Van der Veldt, C. de Laat, R. Hofman and H. Bal, "Profiling energy consumption of VMs for green cloud computing," in *IEEE 9th International conference on Dependable, Autonomic and Secure Computing (DASC)*, Sydney, Australia, 2011.

[7] A. Kansal, F. Zhao, J. Liu, N. Kothari and A. A. Bhattacharya, "Virtual Machine Power Metering and Provisioning," in *1st ACM Symposium on Cloud Computing (SOCC)*, Cascais, Portugal, 2011.

[8] A. E. H. Bohra and V. Chaudhary, "Vmeter - Power modelling for virtualized clouds," in *IEEE International Symposium on Parallel & Distributed Processing (IPDPSW)*, Atlanta, Georgia, USA, 2010.

[9] J. Mars and L. Tang, "Whare-map: heterogeneity in "homogeneous" warehouse-scale computers," in *40th ACM/IEEE International Symposium on Computer*

Architecture (ISCA), Tel-Aviv, Israel, 2013.

[10] L. Bo, L. Jianxin, H. Jinpeng, W. Tianyu, L. Qin and Z. Liang, "EnaCloud: an Energy-saving Application Live Placement Approach for Cloud Computing Environments," in *IEEE International Conference on Cloud Computing*, Los Angeles, CA, USA, 2009.

[11] J. L. March, J. Sahuquillo, S. Petit, H. Hassan and J. Duato, "Real-Time Task migration with Dynamic Partitioning to Reduce Power Consumption," in *XXII Jornadas del Paralelismo*, La Laguna, Spain, 2011.

[12] K. Ye, D. Huang, X. Jiang, H. Chen and S. Wu, "Virtual Machine Based Energy-Efficient Data Center Architecture for Cloud Computing: a Performance Perspective," in *International Conference on Green Computing and Communications*, Hangzhou, China, 2010.

[13] L. Lu, H. Zhang, E. Smirni, G. Jiang and K. Yoshihira, "Predictive VM Consolidation on Multiple Resources: Beyond Load Balancing," in *21st International Symposium on Quality of Service*, 2013.

[14] W. Deng, F. Liu, H. Jin, X. Liao, H. Liu and L. Chen, "Lifetime or Energy: Consolidating Servers with Reliability Control in Virtualized Cloud Datacenters," in, 2012.," in *4th International Conference on Cloud Computing Technology and Science*, Taipei, Taiwan, 2012.

[15] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers," *Concurrency and Computation: Practice and Experience,* vol. 24, no. 13, pp. 1397-1420, 2012.

[16] J. Mars, L. Tang, R. Hundt, K. Skadron and M. L. Sofia, "Bubble-up - Increasing Utilization in Modern Warehouse Scale Computers via Sensible Co-locations," in *44th IEEE/ACM Symposium on Microarchitecture (MICRO'44)*, Porto Alegre, Brazil, 2011.

[17] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen and C. Pu., "An analysis of performance interference effects in virtual environments.," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, San Jose, CA, USA, 2007.

[18] G. Kousiouris, T. Cucinotta and T. Varvarigou, "The effects of scheduling, workload type and consolidation scenarios on virtual machine performance and their prediction through optimized artificial neural networks," *Journal of Systems and Software,* vol. 84, no. 8, pp. 1270-1291, 2011.

[19] G. Somani and S. Chaudhary, "Application Performance Isolation in Virtualization," in *IEEE International Conference on Cloud Computing (CLOUD'09)*, De Pere, WI, USA, 2009.

[20] J. Yang, X. Zhou, M. Chrobak, Y. Zhang and L. Jin, "Dynamic Thermal Management through Task Scheduling," in *IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS'08)*, Austin, TX, USA, 2008.

[21] D. Vanderster, A. Baniasadi and N. Dimopoulos, "Exploiting Task Temperature Profiling in Temperature-Aware Task Scheduling for Computational Clusters," in *12th Asia-Pacific conference on Advances in Computer Systems Architecture (ACSAC'07)*, Seoul, Korea, 2007.

[22] C. Li, R. Zhou and T. Li, "Enabling Distributed Generation Powered Sustainable High-Performance Data Center," in *International Symposium on High-Performance Computer Architecture (HPCA)*, Shenzen, China, 2013.

[23] I. Goiri, K. Le, M. E. Hague, R. Beauchea, T. D. Nguyen, J. Guitart, J.Torres and R. Bianchini, "GreenSlot: scheduling energy consumption in green datacenters," in *International Conference for High Performance Computing, Networking, Storage and Analysis (SC'11)*, Seattle, WA, USA, 2011.

[24] S. K. Garg, C. S. Yeo, A. Anandasivam and R. Buyya, "Environment-conscious scheduling of HPC applications on distributed Cloud-oriented data centers," *Journal of Parallel and Distributed Computing,* vol. 71, no. 6, pp. 732-749, 2011.

[25] J. M. Pierson, "Green Task Allocation: Taking into account the ecological impact of task allocation in cluster and clouds," *Journal of Green Engineering,* vol. 1, no. 2, pp. 129-144, 2011.

[26] E. Elmroth, J. Tordsson, F. Hernández, A. Ali-Eldin, P. Svärd, M. Sedaghat and W. Li, "Self-management challenges for Multi-cloud architectures," in *4th European conference on Towards a service-based Internet*, Poznan, Poland, 2011.

[27] A. Beloglazov and R. Buyya, "OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds," *Concurrency and Computation: Practice and Experience,* p. (online) DOI: 10.1002/cpe.3314, 2014.

[28] D. Bonde, "Techniques for Virtual Machines Placement in Clouds," Department of Computer Science and Engineering, Indian Institute of Technology, Bombay, India,

2010.

[29] J. Xu and F. Forbes, "Multi-objetive virtual machine placement in virtualized data center environments," in *2010 IEEE/ACM Int'l Conference on Green Computing and Communications (GreenCom)*, Hangzhou, China, 2010.

[30] "APDEX. Application Performance Index," [Online]. Available: http://www.apdex.org. [Accessed November 2014].

[31] S. Song, R. Ge, X. Feng and K. W. Cameron, "Energy profiling and analysis of the HPC Challenge benchmarks," *International Journal of High Performance Computing Applications,* vol. 23, no. 3, pp. 1-12, 2009.

[32] D. Meisner, C. M. Sadler, L. A. Barroso, W.-D. Weber and T. F. Wenisch, "Power Management of Online Data-Intensive Services," in *The 38th Annual Symposium on Computer Architecture (ISCA'11)*, San Jose, California, USA, 2011.

[33] S. Nanda and T.-c. Chiueh, "A survey of virtualization technologies," Department of Computer Science, Stony Brook University, Stony Brook, NY, USA, 2005.

[34] C. Delimitrou and C. Kozyrakis, "iBench: Quantifying interference for datacenter applications," in *2013 IEEE International Symposium on Workload Characterization (IISWC)*, 2013.

[35] "CloudSuite benchmark suite," [Online]. Available: http://parsa.epfl.ch/cloudsuite/cloudsuite.html.

[36] "Stress-ng," [Online]. Available: http://kernel.ubuntu.com/~cking/stress-ng/.

[37] "Sysbench," [Online]. Available: https://launchpad.net/sysbench.

[38] I. Mersenne Research, "Great Internet Mersenne Prime Search," [Online]. Available: http://www.mersenne.org/.

[39] "Pmbw - Parallel Memory Bandwidth benchmark / measurement," [Online]. Available: http://panthema.net/2013/pmbw/.

[40] "Fio - Flexible I/O tester," [Online]. Available: http://git.kernel.dk/?p=fio.git;a=summary.

[41] ESnet Software, "Iperf3: A TCP, UDP, and SCTP network bandwidth measurement tool," [Online]. Available: https://github.com/esnet/iperf.

[42] University of Waikato, "Weka 3: Data Mining Software in Java," [Online]. Available: http://www.cs.waikato.ac.nz/~ml/weka/.

[43] National Aeronautics and Space Agency, "NAS Parallel Benchmarks," [Online]. Available: http://www.nas.nasa.gov/publications/npb.html.

[44] "OptaPlanner," [Online]. Available: http://www.optaplanner.org.

[45] "OptaPlanner construction heuristics and local search configuration document," RedHat, [Online]. Available: https://docs.jboss.org/drools/release/6.0.0.CR5/optaplanner-docs/html_single/.

[46] "Ganglia Monitoring System," [Online]. Available: http://ganglia.sourceforge.net.

[47] "Host-sFlow," [Online]. Available: http://host-sflow.sourceforge.net/.

[48] "The R Project for Statistical Computing," [Online]. Available: http://www.r-project.org/.

[49] E. Oró, N. Pflugradt, M. Macías and D. Nardi, "Deliverable 2.1: Green Management of Data Centres: Hardware and software setup of the semi-virtual micro Data Centre test bench," RenewIT project (FP7 – SMARTCITIES – 2013 - 608679), 2014.

[50] D. Jiankang, W. Hongbo, L. Yangyang and C. Shiduan, "Virtual machine scheduling for improving energy efciency in IaaS cloud," *China Communications,* vol. 11, no. 3, pp. 1-12, 2014.

[51] J. Subirats, "Assessing and forecasting energy and ecological efficiency on Cloud Computing platforms," Technical University of Catalonia. Tech Report UPC-DAC-RR-2013-48, Barcelona, 2013.

[52] "UnixBench - A UNIX benchmark suite," [Online]. Available: https://code.google.com/p/byte-unixbench/.

ANNEX A: BENCHMARKING VALIDATION

Comparison of correlation and Mean Absolute Percentage error for different architectures and validation benchmarks, when comparing the values of the estimated power values as output from the power models with the measured values.

AMD Opteron 6140		
Benchmark	**Correlation**	**MAPE %**
Data Analytics	0,730	7,970
Data Caching	0,974	7,520
Data Serving	6,133	0,680
Graph Analytics	0,941	7,696
Media Streaming	0,848	5,081
Software Testing	0,987	1,960
Web Search	0,986	3,823
Web Serving	0,743	11,797
NAS	0,986	3,337

Intel X5650		
Benchmark	**Correlation**	**MAPE %**
Data Analytics	0,663	3,822
Data Caching	0,985	2,878
Data Serving	0,493	2,629
Graph Analytics	0,916	4,740
Media Streaming	0,917	4,494
Software Testing	0,985	3,561
Web Search	0,967	3,923
Web Serving	0,453	2,015
NAS	0,988	2,245

Intel E5-2650		
Benchmark	**Correlation**	**MAPE %**
Data Analytics	0,628	7,777
Data Caching	0,941	4,257
Data Serving	0,603	2,589
Graph Analytics	0,931	6,151
Media Streaming	0,932	3,941
Software Testing	0,945	5,857
Web Search	0,974	20,019
Web Serving	0,527	4,240
NAS	0,963	5,755

AMD Opteron 6234		
Benchmark	**Correlation**	**MAPE %**
Data Analytics	0,757	6,019
Data Caching	0,945	4,894
Data Serving	0,839	8,422
Graph Analytics	0,952	6,251
Media Streaming	0,965	4,806
Software Testing	0,986	3,883
Web Search	0,988	9,201
Web Serving	0,956	9,870
NAS	0,955	5,359

Intel E5-2640		
Benchmark	**Correlation**	**MAPE %**
Data Analytics	0,898	8,313
Data Caching	0,872	3,656
Data Serving	0,644	4,986
Graph Analytics	0,924	2,887
Media Streaming	0,979	4,222
Software Testing	0,957	6,608
Web Search	0,959	1,484
Web Serving	0,903	2,695
NAS	0,983	3,974

ANNEX B: SYSTEM-LEVEL RESOURCES

The models that have been released in deliverable 2.2 consider the next system-level resources, as output of the monitoring system:

- *powerWatts*: measured power (in watts) for a computing node
- *contexts*: CPU context switches
- *instructions*: executed instructions
- *cpu-cycles*: CPU cycles
- *cpu-migrations*: migrations of processes between CPUs
- *branches*: branch-type instruction executions
- *branch-misses*: misses of the CPU branch predictor
- *L1-icache-load-misses*: misses of the Level-1 instruction cache
- *L1-dcache-loads*: loads of the Level-1 data cache
- *L1-dcache-load-misses*: load misses for the Level-1 data cache
- *L1-dcache-stores*: stores for the Level-1 data cache
- *L1-dcache-store-misses*: store misses for the Level-1 data cache
- *LLC-loads*: loads of the Last-Level Cache
- *LLC-load-misses*: load misses of the Last-Level Cache
- *LLC-stores*: stores of the Last-Level Cache
- *LLC-store-misses*: store misses of the Last-Level Cache
- *SIMD_FP_256_PACKED_SINGLE*: single-precision, floating-point, Single-Instruction Multiple-Data (SIMD) executed instructions
- *SIMD_FP_256_PACKED_DOUBLE*: double-precision, floating-point, SIMD executed instructions
- *FP_COMP_OPS_EXE_X87*: Floating Point instructions executed by the coprocessor
- *FP_COMP_OPS_EXE_SSE_FP_PACKED_DOUBLE*: number of double-precision, floating-point instructions, vector-oriented, instructions executed by the Streaming SIMD Extensions (SSE) unit
- *FP_COMP_OPS_EXE_SSE_FP_SCALAR_SINGLE*: single-precision, floating-point, scalar-oriented, instructions executed by the SSE unit

- *FP_COMP_OPS_EXE_SSE_PACKED_SINGLE*: single-precision, vector-oriented micro-operations of the SSE unit
- *FP_COMP_OPS_EXE_SSE_SCALAR_DOUBLE*: double-precision, scalar-oriented, micro-operations of the SSE unit
- *UOPS_RETIRED_ALL*: retired micro-operations
- *Core_0, … Core_N-1*: for a N-cores host, the load from core 0 to core N-1
- *Core_1CPU:* number of CPUs with only 1 active thread
- *Core_2CPU*: number of CPUs with two active threads (hyperthreading)
- *numsockets:* number of active sockets
- *bytes_read*: read bytes into disk
- *bytes_written*: written bytes into disk
- *disk_free*: free disk
- *disk_total*: total disk
- *part_max_used*: maximum partition space used
- *read_time*: average read time into disk
- *write_time*: average write time into disk
- *reads*: number of disk reads
- *writes*: number of disk writes
- *mem_buffers*: number of memory buffers
- *mem_cached*: size of the cached memory
- *mem_free*: free memory
- *mem_shared:* shared memory between processes
- *page_in*: pages of the process inside the main memory
- *page_out*: pages of the process outside the main memory (swap memory)
- *swap_free*: free swap memory
- *swap_in*: number of times the entire process memory has been written into swap
- *swap_out*: number of times the entire process memory has been written from swap
- *bytes_in, bytes_out*: input/output bytes of the network
- *errs_in, errs_out*: input/output network errors
- *pkts_in, pkts_out*: input/output network packets

- *cpu_idle*: CPU idle time
- *cpu_intr, cpu_sintr*: CPU time executing hardware and software interruptions, respectively
- *cpu_nice*: CPU time occupied by processes with a positive nice value
- *cpu_system*: CPU time occupied by the operating system
- *cpu_user*: CPU time occupied by the user process
- *cpu_wio*: CPU time waiting for Input/Output operations
- *interrupts*: CPU interruptions