Measuring and Analyzing Energy Consumption of the Data Center

Introduction

The evolution of cloud computing which provides on-demand provisioning of elastic resources with pay-as-you-go model has transformed the Information and Communication Technology (ICT) industry. Over the last few years, large enterprises and government organizations have migrated their data and mission-critical workloads into the cloud. As we are moving towards the fifth generation of cellular communication systems (5G), Mobile Network Operators (MNO) need to address the increasing demand for more bandwidth and critical latency applications. Thus, they leverage the capabilities of cloud computing and run their network elements into distributed cloud resources.

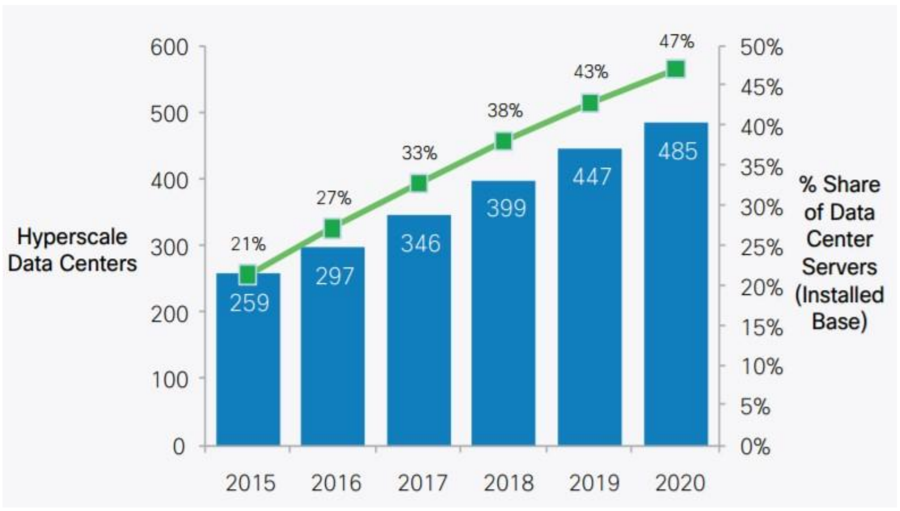


Fig. 1 Growth of hyperscale data centers by 2020 [1].

The adoption of cloud computing by many industries has resulted in the establishment of humongous data centers around the world containing thousands of servers and network equipment. Data centers are large-scale physical infrastructures that provide computing resources, network and storage facilities. Cloud computing is expanding across different industries and along with it the footprint of data center facilities which host the infrastructure and run the services is growing. Since 2015 there has been 259 hyperscale data centers around the globe, and by 2020 this number will grow to 485 as shown in Fig. 1. These type of data centers will roughly accommodate 50% of the servers installed in all the distributed data centers worldwide [5].

Data centers are promoted as a key enabler for the fast-growing Information Technology (IT) industry, resulted a global market size of 152 billion US dollars by 2016 [2]. Due to the big amount of equipment and heavy processing workloads, they consume huge amount of electricity resulting in high operational costs and carbon dioxide (CO_2) emissions to the environment. In 2010, the electricity usage from data centers was estimated between 1.1% and 1.5% of the total worldwide usage, while in the US the respective ratio was higher. Data centers in US consumed 1.7% to 2.2% of the whole US electrical usage [3]. Fig. 2 shows that over the past few years, data center's energy consumption increases exponentially.

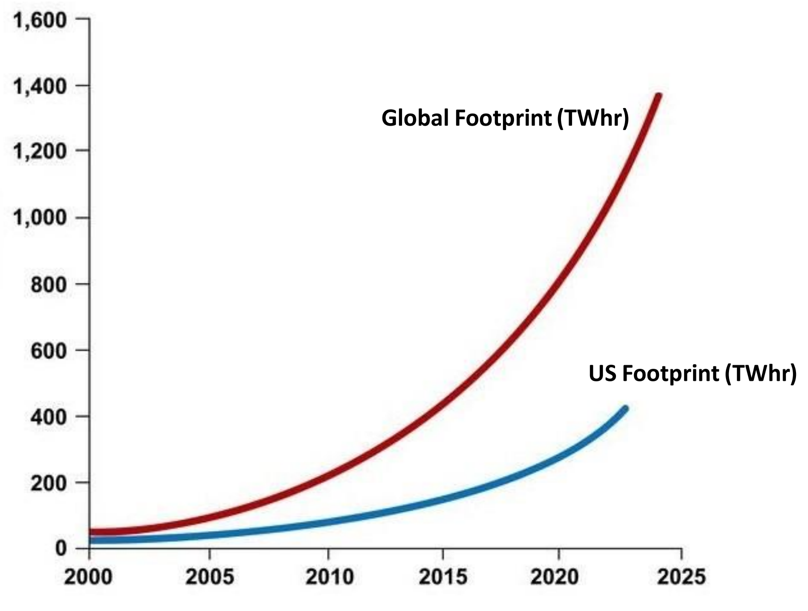


Fig. 2 Projection of data centers' electricity usage [4].

1.1. Objective of the thesis

As discussed in the previous section, the number of data centers increases and so does their respective electricity usage. There has been increased interest from data center's vendors and operators as well as from the academia, to understand how the energy is consumed among different parts of the data center’s infrastructure. A wide body of the research is focused on modeling and prediction of energy consumption. The majority of studies [2] is focused on the energy consumption of the computing subsystems such as servers, network plane and storage, while there are also studies on the energy usage from mechanical and electronic subsystems such as computer room air conditioning (CRAC) units.

This work is focused on the energy consumption of the server, which belongs to the data center’s IT infrastructure. Understanding how the energy is consumed among the components of the server is essential to define an energy prediction model. This thesis presents a system-level power model of the server, which includes the power consumption of the processor, the random-access memory (RAM), and the network interface controller (NIC).

The model is based on Lasso linear regression [67] with non-negative coefficients. The selected variables reflect the power consumption activity of each hardware component. The model is independent of the usage of the server. The test cases are created in a way to explore all the possible workloads of each hardware component. The data collection includes the regression variables as well as the power consumption associated with each measurement. We used regular approach for dividing our data set into training, testing and validation sets. The model which is derived after fitting the data with the training set, is tested for its prediction accuracy against the testing set. The validation set which is a random sample of the data collection, is used afterwards to evaluate our predictions against data which is unknown to our prediction model.

This study, demonstrates the feasibility of deriving a system level power model that can be applied for predicting the energy consumption of the server without using any available tool or utility. We show the advantage of using L1 regularization for reducing the number of coefficients in regression-based power modeling. We also provide a power model that is independent of power usage scenarios and which can be used for run time power estimation with reasonable accuracy.

1.2. Structure of the thesis

The thesis is structured into five main chapters starting with the introduction of the topic. The rest of the work is organized as follows:

Chapter 2 gives an introduction on data centers energy consumption, and briefly presents the architecture of data center. It explains the importance of energy consumption modeling and prediction, and describes how they are used to increase the energy efficiency in different areas of data center such as computing resources, network plane, virtualization layer and associated business cases. Finally, it presents power modeling and prediction approaches in processor, server and data center levels.

Chapter 3 presents the regression based power model that we use to construct the equation which predicts the energy consumption of the server. It describes the regression variables that are used during the model fitting and explains the methodology followed for deriving the model.

Chapter 4 describes how the experiment is set up. It presents the characteristics of the server which is used for data collection, and the tools that we developed and used to collect the data. The test cases which are created to cover all possible workloads are also presented in this section. Then it explains the steps which are done for fitting the model and presents the final power model. Finally, it explains the model evaluations, and illustrates the results for the accuracy of the model in different server's workloads.

Chapter 5 summarizes the work briefing the purpose of constructing the energy model for the server. It states few observations about the regression variables that are used and the results that we got. Finally, it and proposes alternative directions that could result in more accurate predictions and would test the adaptability of the model in different types of servers.

Chapter 2

Background and Related Work

2.1. Data center energy consumption

Data centers typically are powered by electricity. However, following the strategy for decreasing carbon emissions and complying with sustainable operational models, modern data centers use alternative energy sources such as geothermal, wind and solar power. The electric power flows from external power grids into internal infrastructure facilities, Information Technology (IT) equipment and other support systems. The energy flows to the internal IT facilities through Uninterrupted Power Supplies (UPS) to maintain a consistent power distribution even during possible power failures.

The architecture of a data center is complex since it does not only consist of the hardware elements but also the software that runs in the IT infrastructure. Therefore, we can categorize its elements into two layers which are hardware and software, as shown in Fig. 3. The hardware consists of many components. The major ones are cooling systems, power distribution units, lighting equipment, servers and networking equipment. The software layer can be further divided into two subcategories, the Operating System/Virtualization layer and the applications. The first mainly refer to the host OS that is installed in the servers and the cloud deployment running on top of it. The second refers to the different type of applications running in the servers which vary depending on the industry and business cases.

Understanding how the energy is shared among the elements of such a complex system as well as predicting energy consumption, requires a system optimization cycle as presented by M. Dayarathna *et al*. in [6]. Whether we want to model the consumption of the whole data center or we are particularly interested in the IT infrastructure, the general approach can be narrowed down to the following process. Initially, we need to measure the energy consumption of each component that is considered and identify where the most energy is consumed. For that we select the features that will construct the power model. Different techniques can be used for feature selection, such as regression analysis and machine learning. The accuracy of the power model needs to be validated. Finally, the model can be used to predict the system's energy consumption, and find out means to focus on improving the energy efficiency of the data center.

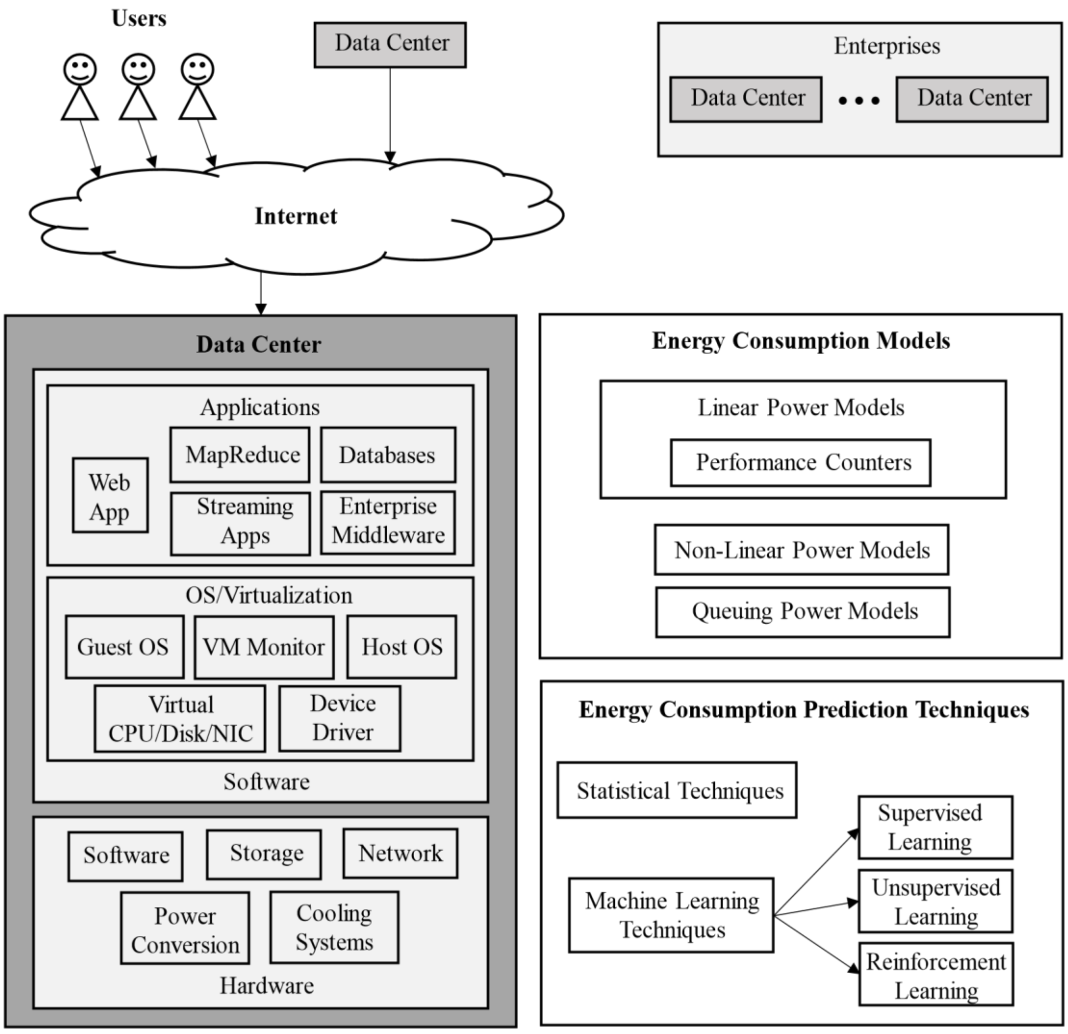


Fig. 3 A view in the context of energy consumption modeling and prediction in data centers. The components of a data center can be categorized into two main layers: software and hardware.

There has been a lot of research on energy consumption prediction for data centers Information Technology infrastructure. Initiatives and efforts to reduce the cost associated with the power distribution and cooling of the equipment are expanding, hence the power management has become an essential issue in enterprise environments. Server takes the highest chunk of the energy consumption in the racks [6]. Consequently, there is need of understanding how the energy is consumed and what are the main components of the server which impacts the energy consumption the most. Several studies [57][59] evaluate the system level power consumption and propose models which predict the energy considering various server components. Other studies [43][50][54][55] focus on processor's power consumption. The processor is the component inside the server which consumes the most energy.

Hardware Performance Counters are proposed in many works [43][50][55] to estimate power consumption of any processor with the use of data analytics or statistical techniques. They provide significant information about the performance of the processor. The event counter mechanisms have different implementation which varies depending on the processor family, and so does the number of the available software and hardware events. Also, there are limitations on how many events can be measured simultaneously. For example, the IBM Power 3-II has 238 available performance counters, while only 8 of them can be measured simultaneously [7]. In the Intel Pentium II processor, only 2 events out of 77 can be measured concurrently [7].

2.2. Towards a green data center

There are many best practices and guidelines for achieving energy efficiency in the data centers. Nevertheless, the over-provisioning of IT resources leads to underutilized IT equipment and energy inefficiency. It is accepted that low utilization of servers, inefficient network plane management, limited virtualization adoption, and lack of business models [8] are the most significant factors that impact negative the energy usage cause by IT loads. In this context, we present solutions which try to tackle these challenges and make data centers greener.

2.2.1. Server Plane

Google stated that a typical cluster of servers is utilized on average 10% to 50% [9]. Dynamic power management (DPM) has been proposed to address the energy inefficiency cause by underutilized servers. Dynamic voltage and frequency scaling (DVFS) is one technique for DPM which uses low voltage supply and low frequency. When there is not intensive work load in the server, neither is need to operate a processor at maximum performance [10]- [12] we can apply DFVS to save energy. Another approach is server consolidation. We consolidate jobs to a limited number of highly utilized servers, and switch the rest into low power or OFF states [13]- [17]. Job scheduling is also used to improve workload management focusing on energy conservation to achieve higher server utilization [18]- [20].

2.2.2. Network Plane

These studies [21]- [23] estimate that network infrastructure consumes 20% to 30% of the total energy consumption in a data center. Network links are highly underutilized, operating between 5% and 25% [24], and they remain idle for approximately 70% of the time [25]. K. Bilal et al. [26], explain that the conventional three-layer tree topology consumes significant amount of energy, partially because enterprise level devices consume a lot of electricity. Alternative data center topologies, such as Fat-Tree [20], Jellyfish [28] and VL2 [29], are proposed to tackle the energy inefficiency of the conventional topologies.

Virtualization techniques is another approach for better network resource utilization. Software defined networking (SDN) solutions provide dynamic resource allocation [30], and network resource scalability by adjusting the active network components of the data center [31]. Network load and energy consumption proportionality is another factor that impacts the energy consumption in the data center networks. Network devices remain underutilized; however, they consume significant amount of energy. D. Abts et al. [32], present methods that adjust the power consumption and the performance of the network based on the amount of traffic. C. Gunarante et al. [33], show that Adaptive Link Rate (ALR) in Ethernet links can maintain a lower data rate for more than 80% of the time, saving significant energy and only adding a very small inflation to the delay. The conventional tree topology or the strictly adoption of bisection topologies such as VL2, Fat-Tree and Jellyfish, are not separate solutions towards an energy efficient network plane. Data centers run different type of applications and services.

A holistic design and common approach for a specific topology does not benefit neither the quality of services nor the operation of the data center. There is need for application-specific solutions that suit the dynamics of the modern data centers and allow flexibility in energy utilization management, improving the efficiency [34]. SDN and virtualization techniques are undoubtedly game changers when reshaping and optimizing the network infrastructure. These technologies allow us to adjust the operation of the network plane based on the network load which yields remarkable amount of energy consumption. Disruptive evolution of data analytics and machine learning can provide meaningful information to the data center operators. They can predict the traffic and adjust the behavior of the network elements according to the demand, without compromising the quality of service (QoS).

2.2.3. Virtualization Plane

One of the benefits of virtualization is that we can share the available physical resources among virtual machines (VMs). Dynamic resource allocation is a technique which allow us to utilize and assign dynamically free computing resources among VMs, saving at the same time considerable amount of energy. Virtualization is an efficient way to provide server consolidation and power off the server that operate in idle mode. In many cases, an idle server consumes 70% of the power consumed by a server running at the full CPU load [35].

2.2.4. Business Model Plane

Lack of proper business models, pricing policies and conflicting priorities, are additional reason for creating energy inefficiency. Customers are not charged in proportion to their resource utilization. The lack of monetary agreements between data center owners and customers increases the interest towards an environmental chargeback model that charges tenants based on their energy consumption [36]. Microsoft has implemented such chargeback models where customers are charged according with their power usage [37].

An alternative approach that can help businesses to adopt energy practices is the collection of metrics related to energy consumption. There are various metrics which allow us to measure infrastructure energy efficiency. The Green Grid consortium has proposed Power Usage Effectiveness (PUE). It defines the ratio of energy that is consumed for cooling and power distribution of the IT infrastructure, to the energy used for computing. PUE closer to 1.0 means nearly all the energy is consumed for computing. Often PUE does not consider all the elements of the IT infrastructure, hence it adequately reflects efficiency [38]. There has been effort to redefine the metrics that measure the energy efficiency in a better context such as power to performance effectiveness (PPE), data center productivity (DCeP), and data center infrastructure efficiency (DCiE) [38], [39].

Architecture of cloud infrastructure and multitenant virtualized environments of data centers increase the complexity of the chargeback models. Resources are shared and therefore chargeback models that incorporate flexible pricing models must be developed. There are initiatives to integrate such models with cloud infrastructure and data analytics. One example is Cloud Cruiser for Amazon Web Services [40].

2.3. Processor level power modeling

Processor is one of the largest power consumers of a server [41]. It has been shown that the server power consumption can be described by a linear relationship between the power consumption and CPU utilization [42]. There are many comprehensive power models that rely on specific details of the processor architecture and achieve high accuracy in terms of processor power consumption model. This section, describes different studies on modeling and predicting power consumption using performance counters and associated events, in processor level. These models, applied during thread scheduling and Dynamic Voltage/ Frequency Scaling (DVFS) configuration.

Rodriges *et al.* [43] estimate power consumption in real time by exploring microarchitecture-independent performance counters. They use two different CPU cores, one suitable for low power applications and another for high performance applications. Their study considers a small set of events and associated counters that have strong impact to the power consumption. These variables are available in both types of cores and are obtained from [44]. Super ESCalar Simulator (SESC) [45] is used for simulating the architectural performance, and Wattch [46] for monitoring the actual value of power consumption. 8 benchmarks are selected from SPEC [47], MiBench [48], and mediabench suites [49], to test the variables for both types of cores. The correlation between each variable and power consumption is computed by using the Pearson's correlation formula. Their results show a high correlation among Fetched instructions, L1 hits, IPC, Dispatch Stalls and retired memory instructions, indicating a power estimation across multiple types of architecture with an average prediction error of 5%.

Singh *et al*. [50] propose Performance Monitoring Counters (PMC) for power consumption estimation using analytic models. They use perfmon utility to collect the counters from an AMD Phenom processor. Based on their correlation to the power consumption, four counters, L2-chache-miss, retired-uops, retired-mmx-and-fpinstructions, and dispatch-stalls, are selected for the experiment. Using piece-wise linear model based on least square estimator, they derive a prediction model which maps observed event rates to CPU core power consumption. Their model is evaluated using NAS [51], SPEC-OMP [52], and SPEC 2006 [53] benchmark suites, and shows an average median error of 5.8%, 3.9% and 7.2% respectively. The model performs run-time, power-aware thread scheduling, to suspend and resume processes based on the power consumption.

Joseph and Martonosi [54] present the Castle project which leverages information from hardware to estimate the actual runtime power consumption in different processor units. In some processors, performance counters do not capture events associated with the power consumption. This work analyzes resource utilization of processor's units by defining a list of heuristic approximations such as number of instruction window physical register accesses, Load/Store Queue (LSQ) physical register access, window selection, and number of wakeup logic accesses. An Alpha 21264 microprocessor is resembled using Wattch power simulation, and a combination of performance counters and heuristic approximations for Wattch-Alpha model is used to approximate some utilization factors. Wattch's basic power model is assumed for the experiments and is analyzed with SPEC95 Int and FP benchmarks. The results show that their heuristic approach estimates the utilization rates within 5% error.

Contreras and Martonosi [55] employ performance counters to estimate the power consumption of CPU and memory of an Intel PXA255 processor. Their linear model uses power weights that map processor and memory power consumption. The model is tested with a set of benchmarks including SPEC2000, Java CDC and Java CLDC programming environments, and gives predictions with an average error of 4%. The model can be applied in a number of settings related to the DVFS configuration environment.

2.4. Server level power modeling

Server is the main component of the data centers IT infrastructure. It runs most of the computational workloads and store all the data. Due to its heavy use, it consumes large amount of the energy, as shown in Fig. 4, and is the most power proportional equipment available in a data center. This section, presents few studies which analyze and model the energy consumption on server level. In the context of our approach, the following studies leverage the data associated with performance counters, to construct power models which are utilized for predicting and optimizing the energy consumption of the server.

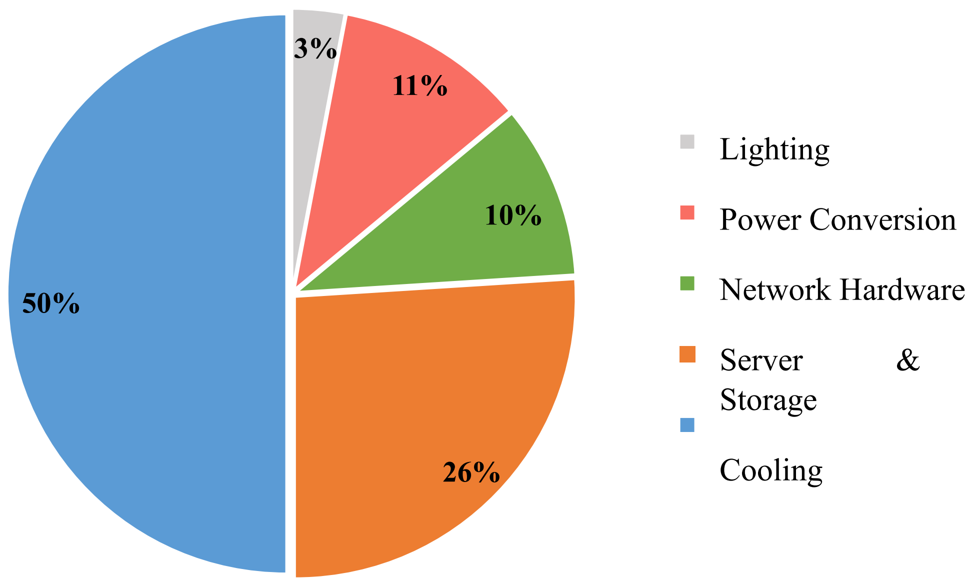


Fig. 4 Distribution of energy consumption in different components of a data center [56].

Bircher and John [57] use Hardware Performance Counters for online measurement of complete system power consumption. They develop power models for microprocessor, access memory, I/O, disk, chipset and Graphics Process Unit (GPU), for two different systems, a quad-socket Intel server and an AMD dual-core with GPU desktop. The two systems are tuned with a set of scientific, commercial and productivity workloads, and selected performance counters events are collected with the Linux *perfctr* [58] device driver. The correlation analysis between the counters and the power consumption is performed using linear and polynomial regression modeling.

Their study shows that events related to the microprocessor have significant correlation to power consumption in the subsystems including memory, I/O, hard drive disk (HDD), chipset and microprocessor, and they can accurately estimate the total system power consumption. Instead of creating a prediction model for the whole server, they use HPCs associated to the processor, to estimate the energy of other subsystems. Moreover, in comparison with our study, they also consider GPU which is not included in our model, because the server we study is not intended for graphical usage neither video or image processing.

Economou *et al*. [59] propose Mantis, a non-intrusive model for real-time server's power estimation. Mantis uses low-overhead OS utilization metrics and performance counters to predict power. It requires a one-time, offline calibration phase to extract basic AC power consumption characteristics and relate them to the system performance metrics. Benchmark is done by individually stressing the major components of the blade system, CPU, access memory, hard disk and network, to derive the basic correlation between their utilization and power consumption. A linear model is used to fit the data relating performance counters to AC power variation. The model is developed for blade and Itanium server and is evaluated using SPECcpu2000 integer and floating-point benchmarks, SPECjbb2000, SPECweb2005, the *streams* benchmark, and matrix manipulation, covering multiple computing domains. The prediction error for Mantis is measured within 10% for most workloads.

While our experiment uses UNIX utilities and custom scripts to stress the components of server in different workloads, Mantis' benchmark runs Generic Application eMUlaTion (GAMUT) [60] to emulate various levels of CPU, memory, network traffic and hard disk. We measure the activity levels of the CPU and RAM using HPCs for fine granularity. Instead, they use Operating System (OS) performance metrics and specifically CPU utilization as variable for measuring the activity of the CPU, while HPCs are used for measuring the impact of RAM.

A.Lewis *et al.* [72] use linear regression to create a power prediction model which dynamically correlates system bus traffic with processor's task activities, RAM metrics, and motherboard power measurements. The accuracy of the model is on average 96%.

The model demonstrates the energy relationship between the workload and the thermodynamics of the server. The regression variables includes the values of the energy consumed by processor, RAM, electromechanical subsystems, motherboard, and HDD. Analysis of variance (ANOVA) [73] is used to calculate the best fit to the data. The validation of the energy model is done using SPEC CPU2006 benchmark suite.

Both Lewis model and our study use HPCs for collecting processor and RAM specific metrics. Nevertheless, there are few major differences. They utilize data that impact the thermodynamics of the server, e.g. ambient temperature, and other electromechanical equipment such as cooling fans and optical drives. However, our study takes into account the incoming and outgoing network packets to consider the impact of network traffic. Their model uses 12 regression variables for prediction with a median error of 4%, whereas we use only 2 variables to estimate power consumption with median error of 5.33%. This is a significant difference when applying mechanisms for real time power estimation.

2.5. Data center level power modeling

The power models described in section 2.3 and 2.4 are focused on modeling the energy consumption of processor and server. Beside these individual components, several studies have proposed power models which analyze and estimate the energy consumption in data center level. When constructing higher level power models for data centers, it is essential to understand how the energy is consumed by large groups of servers and what is the role of data center network to the energy consumption.

There are three different types of power models for a group of servers. These are queuing theory based power models, power efficiency metrics based power models, and others. Also, more parameters need consideration compared to power models for a single server. Time delay and sometimes penalties associated with the setup cost i.e. booting the server on, are just few examples. Gandhi *et al.* [61] and Lent [62] propose power consumption models based on queueing theory, which schedule the processes among the servers depending on their power state. There are power models developed based on the data center's performance metrics. Qureshi *et al.* [63] present an energy model for a group of servers, combining the PUE of the whole data center with the linear power model of a single server presented by Fan *et al.* [64].

When modeling energy consumption caused by the data center network, there are multiple components of the network to consider, such as the network equipment and the topology of the network. In higher level abstraction, the energy cost of network links has significant impact to the power consumption. For example, Heller *et al.* [65] present a model for the total network by accumulating the power consumption of an individual link and the cost of an active network switch. Other studies present power models for the network devices. An extensive model by F. Jalali *et al.* [66] describes the total energy consumption of a switch as an addition of input and output energy to and from the switch, the supply and control energies, and the instantaneous throughput.

2.6. Summary

Sections 2.3, 2.4, 2.5 presented approaches for power modeling in three different levels. First, we presented power models that predict the consumption of processor; the component of the server that impacts the energy the most. Power models in processor level, were applied during threading scheduling and DVFS to optimize the utilization and reduce the energy consumption. Next, we discussed about system level power models that include other essential components of the server, such as RAM, hard disk, NIC and GPU. Related studies have shown that we can derive power models which predict the energy consumption of the server in real time with reasonable accuracy, less than 10% regardless of the server's workload [59]. Studies that described power models associated with data center's power consumption were presented at the final section. In such cases, modeling is done for optimizing the energy efficiency of groups of servers, or is focused on data center network, where alternative network topologies and efficient routing mechanisms are proposed for efficient network plane.

Chapter 3

Regression-based Power Modeling

This chapter presents our approach to define a power model which predicts the energy consumption of the server. The methodology to identify suitable variables which reflect the power characteristics of the server is described in detail. We finally explain what are the statistical factors that are used to evaluate the efficiency of the prediction model.

3.1. Lasso model with non-negative coefficients:

A fine-grained approach is used, to derive the model of the energy consumption. We select 30 variables which reveal the power characteristics of the server. Regression-based analysis is followed to estimate the relationship among these variables. Specifically, linear regression based on Lasso method [67] with nonnegative coefficients, is used to create a power model which takes three hardware components into account; these are the processor, the RAM and the Network Interface Controller (NIC). The coefficients show contribution to the power consumption; hence they are limited to get only positive values.

The Lasso is a linear model which performs L1 regularization, thus effectively reduces the number of estimators. It alters the model fitting process to select a subset of the coefficients for use in the final model. The Lasso estimate (1) minimizes least square penalty with $\alpha||w||_1$ added. The parameter α or alpha in its full form, is a constant which controls the degree of sparsity in the estimated coefficients, and $||w||_1$ is the L1-norm of the parameter vector [67]. Showing strong sparse effects, the Lasso model is suitable for our work where we need to keep only the coefficients which have the most significant contribution to the power consumption.

$$\min_{w} \frac{1}{2n_{samples}} ||X_w - y||_2^2 + \alpha||w||_1 \qquad (1)$$

The methodology we follow to derive the power model is based on the following five steps:

- Define the set of regression variables which reflect the activity levels of the hardware.
- Design the energy benchmark, which stresses the regression variables and explores their behavior for different CPU, memory, and network load.
- Run the energy benchmark and collect the regression variables.
- Create the linear model using Lasso to estimate sparse coefficients.
- Validate the power model against the testing data set, and multiple validation sets which cover our benchmark's scenarios.

The power model for the server is presented based on the linear model (2),

$$f(y_i) = b_0 + \sum_{j=1}^{p} b_j\, g_j(x_{i,j}) \qquad (2)$$

where $g_j(x_{i,j})$ is a preprocessing function of the original values of the features, b_0 is the intercept which is equal to the target variable when all the features are set to zero, and b_j is the coefficient value of each feature. The preprocessing function of the features are presented in the Table . The values of the intercept and the coefficients are calculated during the model fitting.

The quality of the prediction model is measured using the Mean Squared Error (MSE) which is a metric corresponding to the expected value of the squared loss [68].

MSE is calculated as the average of the sum of the squared deviations of the predicted variable, $MSE(y_i, f(y_i))$ (3).

$$S(b_0, \dots, b_j) = \sum_{i=1}^{n} \left(y_i - f(y_i)\right)^2 \qquad (3)$$

where n is the number of samples, y_i is the true value, and $f(y_i)$ is the corresponding predicted value.

3.2. Regression variables

We select 30 regression variables which reflect the power characteristics of the processor, the access memory, and the Network Interface Controller of the server. There are two variables related to the network traffic representing the total number of packets received and transmitted on the network interface. The rest 28 variables are relevant to the CPU processing and the memory access. We call them Hardware Performance Counters (HPCs). These counters represent a fine-grained model of analyzing the CPU and the memory utilization, and they are widely used for measuring power consumption in real time. Table 1 presents the regression variables and their preprocessing functions.

Table 1 Description of regression variables and their preprocessing functions.

Hardware Resources	**Regression Variable $x_{i,j}$**	**Preprocessing Function $g_j(x_{i,j})$**	**Description**

Processor	28 HPC event rates: $x_{i,j}\ (j\epsilon[1..28]) = \begin{cases} \frac{c_{i,1}}{d}, for\ cpu-cycles \\ \frac{c_{i,j}}{c_{i,1}}\ for\ other\ HPCs \end{cases}$ (4) Where $c_{i,1}$ is the increment in cpu-cycles, and $c_{i,j}(i\epsilon[2..n], j\epsilon[2..28])$ is the increment for the other HPCs during the same monitoring period d.	Normalization function: $g_j(x_{i,j}) = \frac{x_{i,j}-mean(x_{i,j})}{standard\ deviation(x_{i,j})}$ (5) The mean and the standard deviation of each variable, are calculated from the data set used for model fitting.	HPCs available on Intel Xeon CPUs E5-2680 v3: CPU cycles, branchinstructions, branchmisses, bus-cycles, cache-misses, cache-references, instructions, ref-cycles, L1-dcacheload-misses, L1-dcacheloads, L1-dcache-stores, L1-icache-load-misses, LLC-load-misses, LLCloads, LLC-storemisses, LLC-stores, branch-load-misses, branch-loads, dTLBload-misses, dTLBloads, dTLB-storemisses, dTLB-stores, iTLB-load-misses, iTLB-loads, node-loadmisses, node-loads, node-store-misses, node-stores.
NIC	Incoming packets (Bytes/s) $x_{i,29} = \frac{c_{i,29}}{d_i}$ Outgoing packets (Bytes/s) $x_{i,30} = \frac{c_{i,30}}{d_i}$	$g_j(x_{i,j}) = x_{i,j}$	ifHCInOctets, ifHCOutOctets

As presented in these works [55], [57], measuring power consumption at high frequencies is impractical. Dynamic Voltage and Frequency Scaling (DVFS) can reduce the power consumption. In this case power simulators are time consuming and they do not reveal accurate information for real time power consumption monitoring, hence they are prone to errors [50], [69]. Nevertheless, HPCs reveal considerable amount of information about power consumption [43]. They monitor events associated to the CPU, memory, I/O and network by counting specific events such as CPU cycles, instructions, cache misses, Translation Lookaside Buffer misses, bytes in/out, and so on. Therefore, the importance of measuring and analyzing HPCs is essential when working with power management models for energy consumption estimation and optimization [70].

Power consumption is the rate of energy consumed over a time period, whereas regression variables have different values. HPCs are raw numbers and the variables associated with the network transmission are measured in bytes. In the first phase, we preprocess all the predictor variables to convert them into rates comparable to the power consumption. The design for preprocessing the regression variables is based on [71]. Specifically, CPU cycles are divided by 10 seconds, which is the sampling rate of the events, therefore we get the CPU cycles per second. The rest of the HPCs are divided by the number of CPU cycles of each sample. This way, we have the rate of each HPC per CPU cycle. For the network metrics, we divide received and transmitted packets by 10 seconds to get the data rate in bytes per second (4). In the second phase, we normalize the data using (5). We first calculate the mean value and the standard deviation for every predictor variable. Then we get the preprocessed values of the variables and we normalize them by subtracting their means and dividing by their respective standard deviation.

Chapter 4

Experimentation

4.1. Data collection

For our experiment, we use Nokia Airframe server D51BP-1U. It is equipped with Intel processor, which has two Xeon CPUs E5-2680 v3 at 2.50 GHz. Each CPU has 12 physical cores and hyperthreading enabled. The Network Interface Controller of the server is Intel 82599ES, 10-Gigabit SFI/SFP + Network Connection. The RAM is DDR4, with size of 128 GB and bandwidth 2133 MHz. The server runs CentOS Linux version 7, and the kernel version is 3.10.0-514.2.2.el7.x86_64.

In our benchmark, we have a *bash* script which collects the HPCs, the network traffic, and the power consumption of the server every 10 seconds. Each run of the script represents an event. We run the script for 91 combinations of CPU, memory and network load. For each combination, we repeat the measurement 30 times and collect 30 events. In total, we collect 2730 events. For each event, we collected simultaneously the energy consumption of the server.

There are few reasons for repeating the experiment 30 times for every load combination. The duration of 10 seconds that we log each event is defined by the *sleep* UNIX utility [74], so one reason is that sleeping time slightly fluctuates few milliseconds in every run. Moreover, due to the increased functionalities such as multi-core systems, multi-CPU systems, multi-level caches, non-uniform memory, multi-threading, pipelining and out-of-order execution [75], modern processors have non-linear CPU utilization. Hence, despite the user-level CPU utilization is maintained in the same level throughout all 30 repetitions, the overall CPU utilization is affected by operating system processes that run constantly and they have variation due to the non-linear CPU behavior.

We use *perf* UNIX utility [76] to collect the HPCs from the processor. The counters that are collected are listed below: branch-instructions, branch-misses, bus-cycles, cachemisses, cache-references, CPU cycles, instructions, ref-cycles, L1-dcache-load-misses, L1-dcache-loads, L1-dcache-stores, L1-icache-load-misses, LLC-load-misses, LLCloads, LLC-store-misses, LLC-stores, branch-load-misses, branch-loads, dTLB-loadmisses, dTLB-loads, dTLB-store-misses, dTLB-stores, iTLB-load-misses, iTLB-loads, node-load-misses, node-loads, node-store-misses, node-stores.

Network metrics related to the total number of packets received and transmitted at the network interface are collected through *snmpwalk* application [77]. We run *snmpwalk* to our server to request ifHCInOctets and ifHCOutOctets counters from the leaf switch that is connected to the server's Network Interface Controller. The ifHCInOctets shows the total number of bytes received on the interface, and the ifHCOutOctets shows the total number of bytes transmitted out of the interface. The values of these counters are measured since the initialization of the network management system.

We execute a *C* program, to create load to the access memory. The program allocates large blocks of memory and fills them with 1. To impose load to the CPU, we run *stress* tool [78] in the server, which spawns workers spinning on the *sqrt()*. A single worker imposes 100% load in a CPU thread. In our case, we have in total 24 physical cores with hyperthreading, hence 48 threads in total. Spawning one worker, we fully utilize one thread. Finally, *iperf* tool [79] is used to generate bulk connection between our server and another one which belongs in the same network. The protocol that we use is TCP.

The energy consumption of the server is collected with *ipmitool* [80] directly from the power supply units (PSU). The server has two PSUs, so we calculated the total power consumption by accumulating the energy output from both.

4.2. Test cases

We create 9 test cases based on different memory loads, from 0 to 100%. For each memory load, we run 8 CPU loads from 0 to 100%. The test cases are designed in such a way to cover scenarios for all the range of memory and CPU load of the server. The size of the access memory is 128 GB. Therefore, we have one test case when memory is not stressed. In this case, the only memory overhead is approximately 1.4 GB, and is produced by the system processes of the operating system. Then we have 7 more test cases, running our *C* program which stresses the memory up to maximum with intervals of 16 GB.

For each memory case, we have CPU load increasing from nearly 0% when the server is in idle state, up to 100% when running the *stress* tool and creating maximum CPU load. The intervals of the CPU load increase every 12.5%. Essentially, every time we increase the CPU load by 12.5%, we spawn 6 additional workers.

Table 2 Description of the workloads used in our energy benchmark.

Category	**Description**	**Test Case**	**Id**
CPU	CPU load from 0% to 100%	Keep the RAM idle, while stressing the CPU in 9 workloads from idle to maximum load.	1
RAM	RAM: 17.4 GB CPU load: 0% - 100%	Allocate 16 GB of memory, and stress CPU in 9 workloads from idle to maximum load.	2
RAM	RAM: 33.4 GB CPU load: 0% - 100%	Allocate 32 GB of memory, and stress CPU in 9 workloads from idle to maximum load.	3

RAM	RAM: 49.4 GB CPU load: 0% - 100%	Allocate 48 GB of memory, and stress CPU in 9 workloads from idle to maximum load.	4
RAM	RAM: 65.4 GB CPU load: 0% - 100%	Allocate 64 GB of memory, and stress CPU in 9 workloads from idle to maximum load.	5
RAM	RAM: 81.4 GB CPU load: 0% - 100%	Allocate 80 GB of memory, and stress CPU in 9 workloads from idle to maximum load.	6
RAM	RAM: 97.4 GB CPU load: 0% - 100%	Allocate 96 GB of memory, and stress CPU in 9 workloads from idle to maximum load.	7
RAM	RAM: 113.4 GB CPU load: 0% - 100%	Allocate 112 GB of memory, and stress CPU in 9 workloads from idle to maximum load.	8
RAM	RAM: 126.4 GB CPU load: 0% - 100%	Allocate 125 GB of memory, and stress CPU in 9 workloads from idle to maximum load.	9
Network	CPU idle RAM idle Network: 0 GB/s - 10 GB/s	Keep the CPU and RAM idle, while adding traffic to the server from 0 GB/s to maximum.	10

We also want to evaluate the impact of the network load when there is no added load to the CPU and memory. For this purpose, we have another test case where we set a network connection between the Airframe server and another server with similar specifications and we generate different levels of network traffic between the servers. Both servers have 10-Gigabit NICs, so we simulate 10 traffic scenarios having the network link idle initially, and then increasing the traffic to maximum bandwidth.

4.3. Model fitting

For the model fitting we use Lasso linear model that estimates sparse coefficients. It is inclined to reduce the number of features that contribute to the prediction model. This model is effective in cases we need to select only the few features that have the highest effect in the response variable. The features in our power model contribute positive to the energy consumption, since all of them produce a portion of the total energy. Therefore, we run the Lasso model with non-negative coefficients which forces the coefficients to be positive.

We follow a common data analysis approach when splitting the data set. In the first step of the implementation, we randomly shuffle the events in the data set, and we discard 10%, keeping it separately for validation. With the 90% of the initial data set, we form a new data set, which we split into 70% training and 30% testing data set. Then, we train our model with the training data set. The model tries to learn the effect of each feature to the response variable. At this stage, we get the coefficients of the features that contribute to the power consumption.

Next, we use the 30% of the testing set to predict the values of the power consumption. Along with the predictions, we calculate the MSE which is an estimator of the overall deviations between predicted and expected values. The result for the MSE is 0.02. Additionally, we calculate the variance of the prediction, which is 0.97. The median prediction error is 5.00%. The final power model is presented in the formula (6).

$$\text{Power (W)} = 269.2873 + 67.5317 \times g_1(x_1) + 40.0112 \times g_2(x_2) \quad (6)$$

where $g_i(x_i)$ (i ϵ [1,2]) is the preprocessing function as described in Table . Let c_1 and c_2 be the increment in CPU cycles and dTLB-loads in the monitoring period d, respectively.

$$g_1(x_1) = \frac{x_1 - 61882864724.64}{44874138656.54},\ x_1 = \frac{c_1}{d}$$

$$g_2(x_2) = \frac{x_2 - 0.44}{0.20},\ x_2 = \frac{c_2}{d}$$

The event rate of CPU cycles refers to the clock cycles, and describes the workload of the processor which takes the highest portion of the power consumption. The dTLBloads reflect the efficiency of the access memory and it shows how many times the translation lookaside buffer is accessed in a CPU cycle.

4.4. Model evaluation

We have 2730 events representing our data set. As discussed in section 4.3, 90% of the data equivalent to 2457 events, is used for model fitting, and 10% of the data equal to 273 events, is used for validating the model. The validation data set, which is a random sample of the whole data set, is not used when fitting and testing the prediction model. We run the prediction against the validation set and calculate again the MSE and the variance which are 0.02 and 0.97 respectively. The median error of the validation data set is 5.33%, slightly increased by 0.33% as compared to the median error which is calculated after testing the prediction model against the testing set. Since these two values for median prediction error are very close, we confirm the accuracy of our prediction model.

Furthermore, we analyze the accuracy of our model for all the different workloads to understand its performance in every individual case. These workloads consist of 91 combinations of CPU, memory and network load. We classify the workloads per test case, and calculate their corresponding prediction errors to understand the prediction accuracy for different workloads. The actual value of power consumption is also presented for every workload.

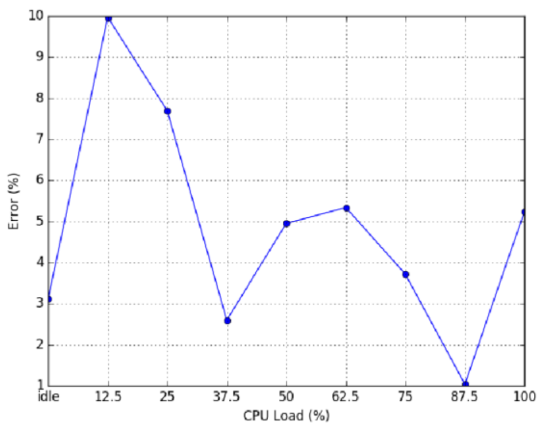


Fig. 5 Percentage of prediction error when RAM is idle, and CPU is stressed from 0% to 100%.

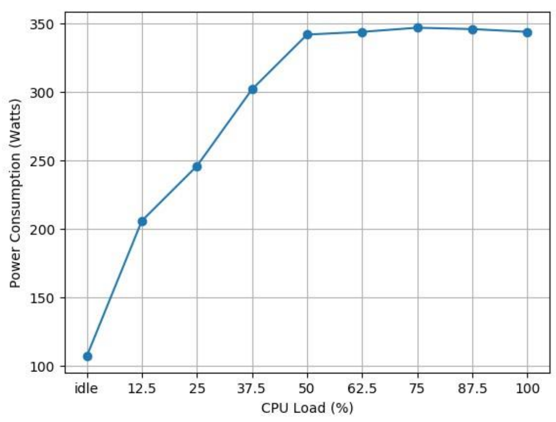
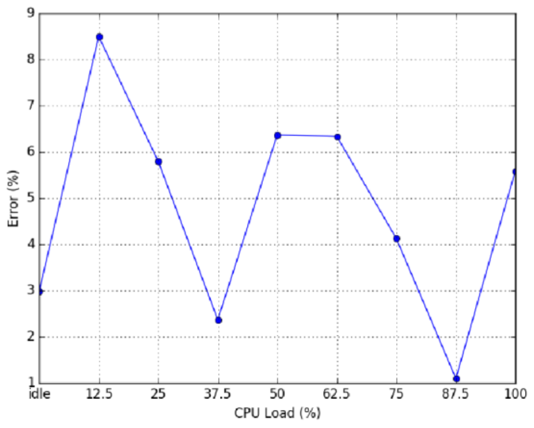


Fig. 6 Power consumption when RAM is idle, and CPU is stressed from 0% to 100%.



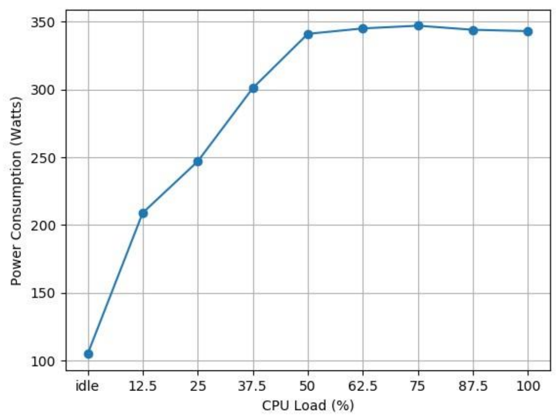
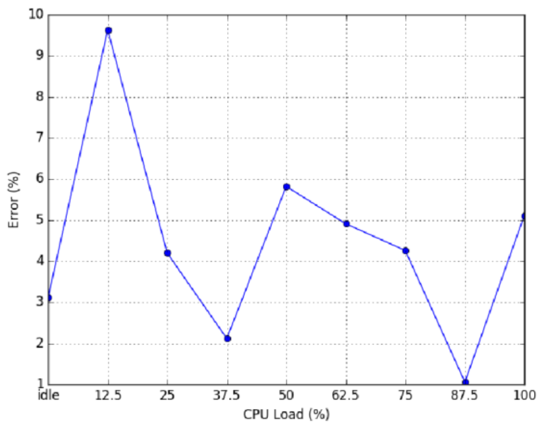


Fig. 7 Percentage of prediction error when 16 GB RAM are allocated, and CPU is stressed from 0% 100%.

Fig. 8 Power consumption when 16 GB of RAM of are allocated, and CPU is stressed from 0% to to 100%.



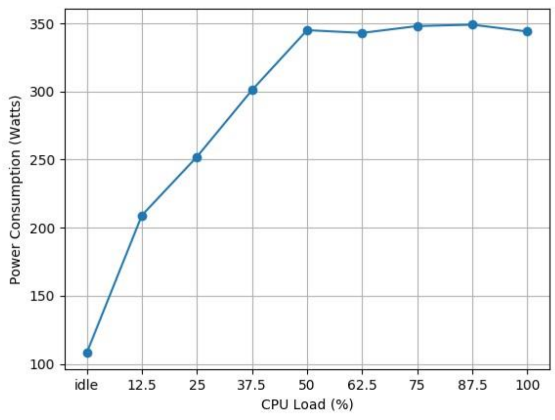
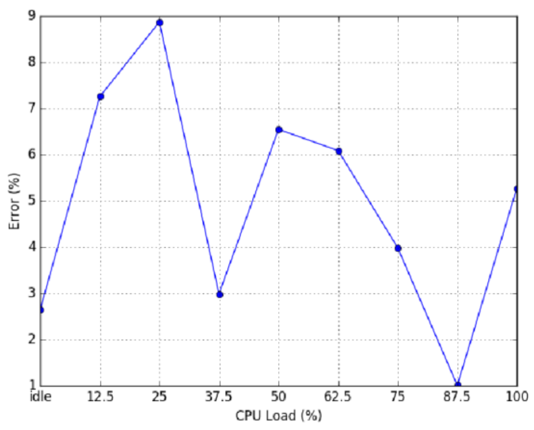


Fig. 10 Power consumption when 32 GB of RAM

Fig. 9 Percentage of prediction error when 32 GB are allocated, and CPU is stressed from 0% to of RAM are allocated, and CPU is stressed from 0%

100%.

to 100%.



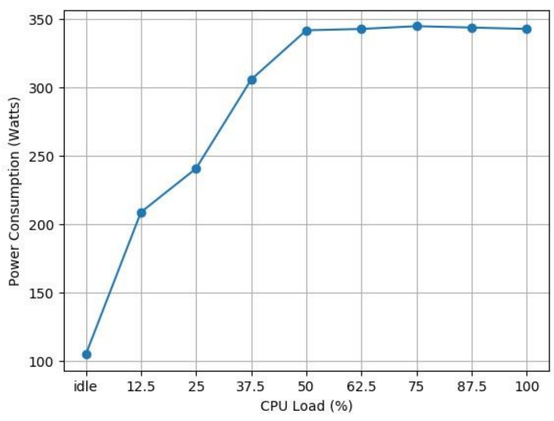
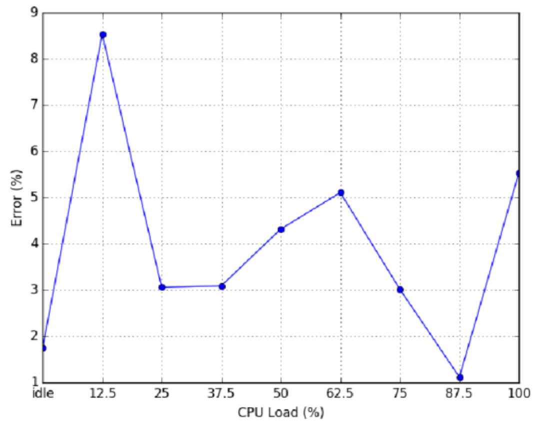


Fig. 11 Percentage of prediction error when 48 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

Fig. 13 Percentage of prediction error when 64 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

Fig. 15 Percentage of prediction error when 80 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

Fig. 12 Power consumption when 48 GB of RAM are allocated, and CPU is stressed from 0% to 100%.



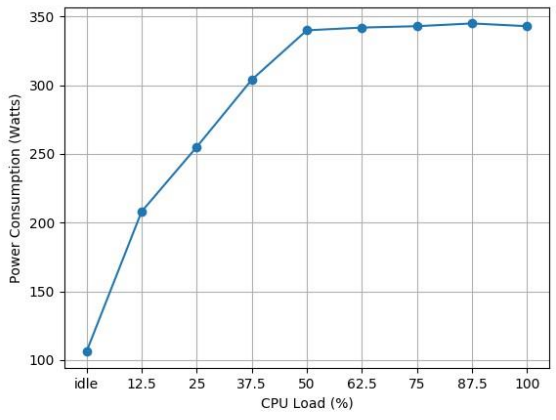
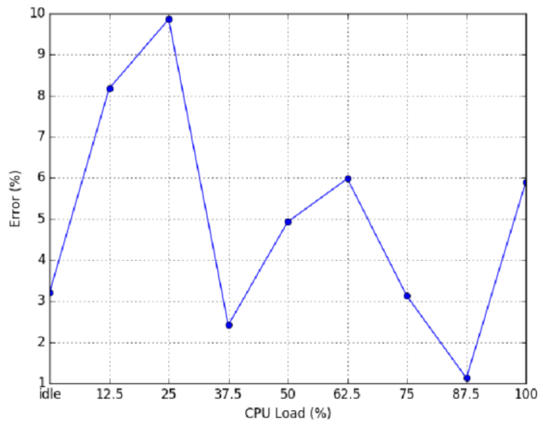


Fig. 14 Power consumption when 64 GB of RAM are allocated, and CPU is stressed from 0% to 100%.



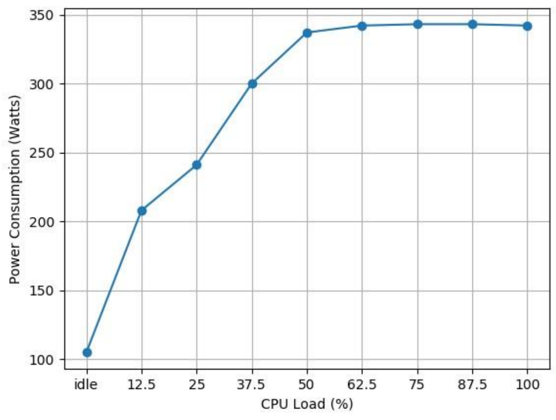
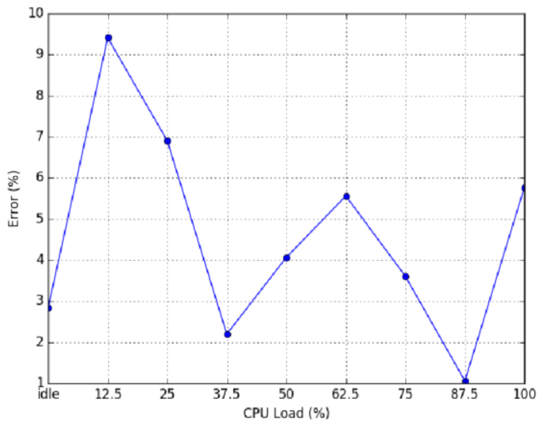
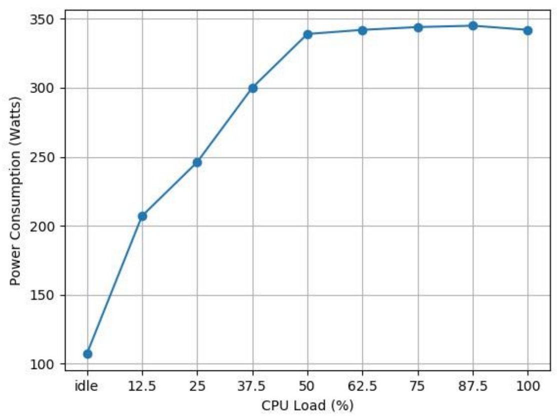
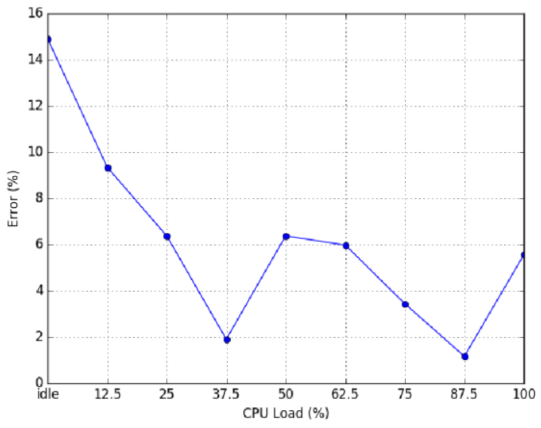


Fig. 16 Power consumption when 80 GB of RAM are allocated, and CPU is stressed from 0% to 100%.







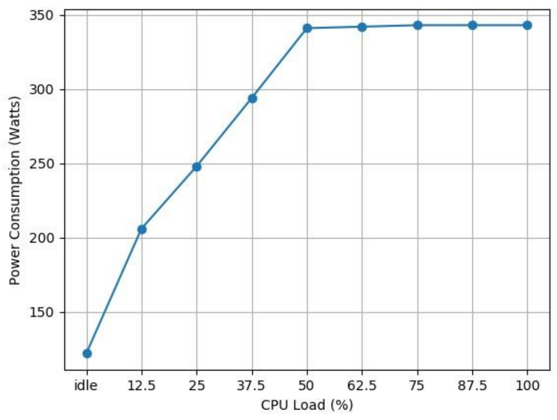


Fig. 17 Percentage of prediction error when 96 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

Fig. 20 Power consumption when 112 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

Fig. 19 Percentage of prediction error when 112 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

Fig. 21 Percentage of prediction error when 125 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

Fig. 18 Power consumption when 96 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

Fig. 22 Power consumption when 125 GB of RAM are allocated, and CPU is stressed from 0% to 100%.

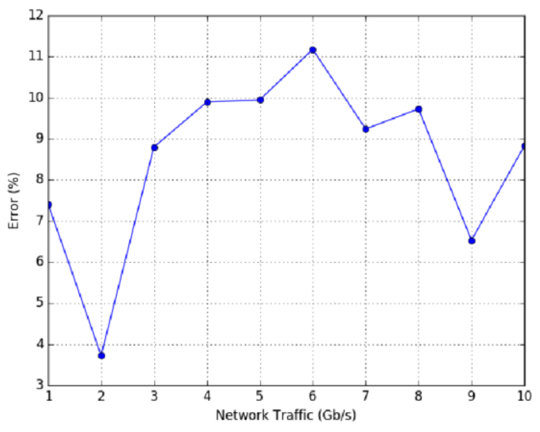


Fig. 23 Percentage of prediction error for different network traffic, when CPU is stressed from 0% to

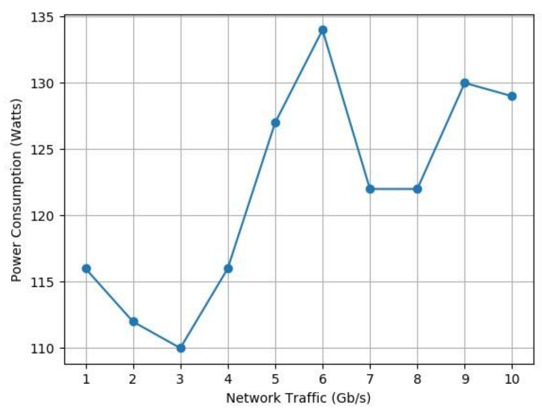


Fig. 24 Power consumption for different network traffic, when CPU is stressed from 0% to 100%. 100%.

Figures 5, 7, 9, 11, 13, 15, 17, 19, 21 and 23 present the results for the behavior of the prediction error for different workloads. Afterwards, we compare them with the median prediction error which is 5.33%. We observe that the smallest error is 1.09% when CPU load is 87.5%, memory is not stressed and there is no added network traffic, while the highest error is 14.76% and is measured when memory is stressed to the maximum level, CPU is idle and there is no added network traffic.

In the test case where network traffic is added, we notice that error is higher than 5.33% for almost all the network loads, but for the case where traffic is set to 2 Gb/s. When CPU is idle, the error is below 5.33% for almost all the memory loads, with an exception when memory is fully stressed, the error exceeds the median error and reaches its pick value of 14.76%. In the case of CPU load 12.5%, for all the memory loads, the error is nearly one and a half to two times higher than the median error, recording values between 6.83% and 11.31%. Error slightly higher than the median error is also measured in every memory case, when CPU is fully utilized. In this case, the maximum variation from the median error is 1.34% for CPU load 100% and 87.5% memory utilization. For the rest of different memory and CPU loads, the prediction error is nearly equal or less than 5.33% in most of the cases Table . Very good predictions of energy consumption with errors between 1.09% and 3.13%, are noticed in the cases of CPU loads 37.5% and 87.5%, for all the memory loads.

Table 3 Median prediction error per test case.

Test Case	1	2	3	4	5	6	7	8	9	10
Error (%)	4.85	5.46	4.47	4.96	3.95	4.31	4.60	4.5	6.12	8.53

Figures 6, 8, 10, 12, 14, 16, 18, 20, 22 and 24 show the actual power consumption of the server for different workloads. Despite the memory utilization, power consumption increases until the CPU load reaches 50%. When CPU load is between 50% and 100% the consumption level remains almost stable, close to its maximum value. In the last case where the impact of network traffic is tested, we notice that power consumption is maintained in low level regardless the amount of incoming and outgoing traffic.

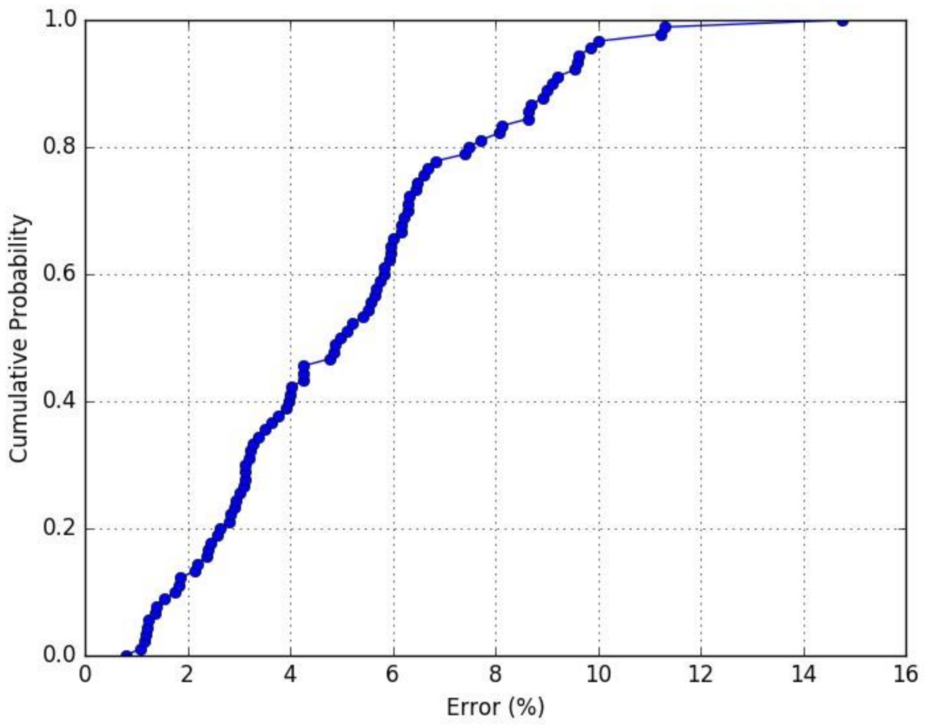


Fig. 25 Empirical Distribution Function

Fig. 25 presents the Empirical Distribution Function of the cumulated median errors for the different combination of CPU, memory, and network load. This is a step function that increases by a factor of 1/91 at each of the 91 median errors. The value of this function at any specified data point represents the fraction of the data points of the median error that are less than or equal to the specified value. For example, we can see that 50% of predictions across all benchmarks have less than 5.33% error.

4.5. Discussion

From Fig. 25 we observe that for the majority of workloads the prediction error lies below 10%. Specifically, for 96.7% of the workloads we can predict the power consumption with an error of less than 9.97%. Three test cases give less accurate predictions with median error per test case between 5.46% and 8.53% Table . These are the one where we allocated 16 GB of RAM, the other where 126 GB of RAM are allocated, and the last one which tests the impact of network traffic to the power consumption. The network test case results the highest median prediction error of 8.53%.

We observe that the features associated with the incoming and the outgoing network traffic show negative correlation with the power consumption. From Fig. 26 and Fig. 27, it is visible that the intersection points of power consumption and network traffic do not follow a linear trendline. Additionally, we calculated the R-squared value for both network features. In case of the incoming traffic the R-squared value is 0.1908, and for the outgoing traffic is 0.1612. The R-squared values reveal that the data associated with the network traffic do not fit a linear regression line, and therefore, we conclude to the fact that network variables are not used in our prediction model.

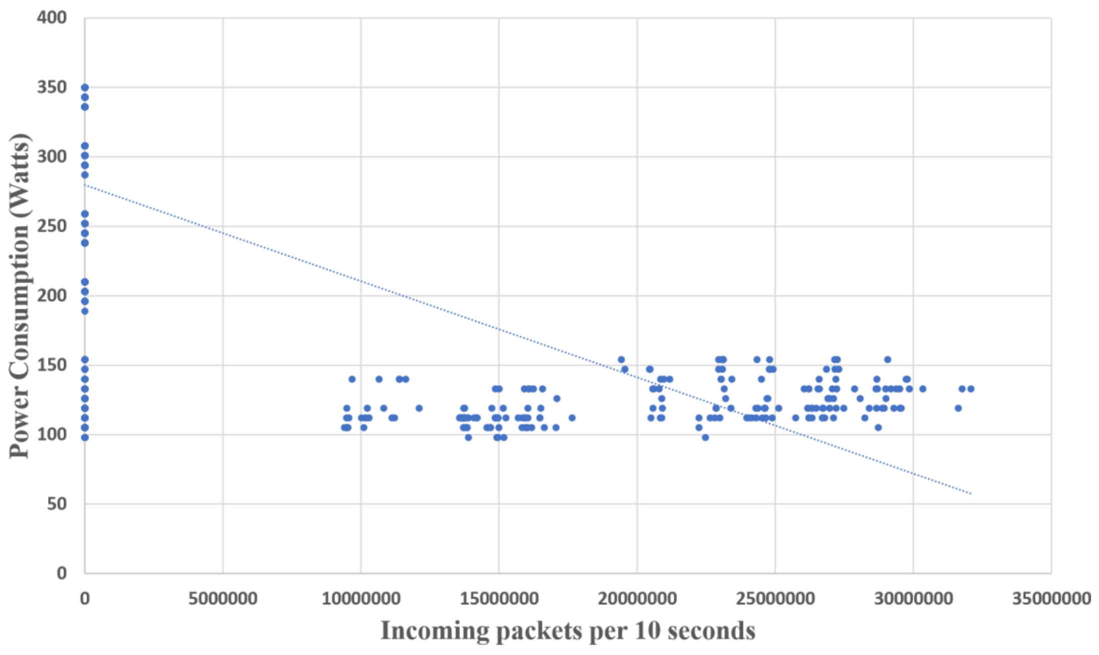


Fig. 26 Correlation diagram between power consumption and incoming network traffic.

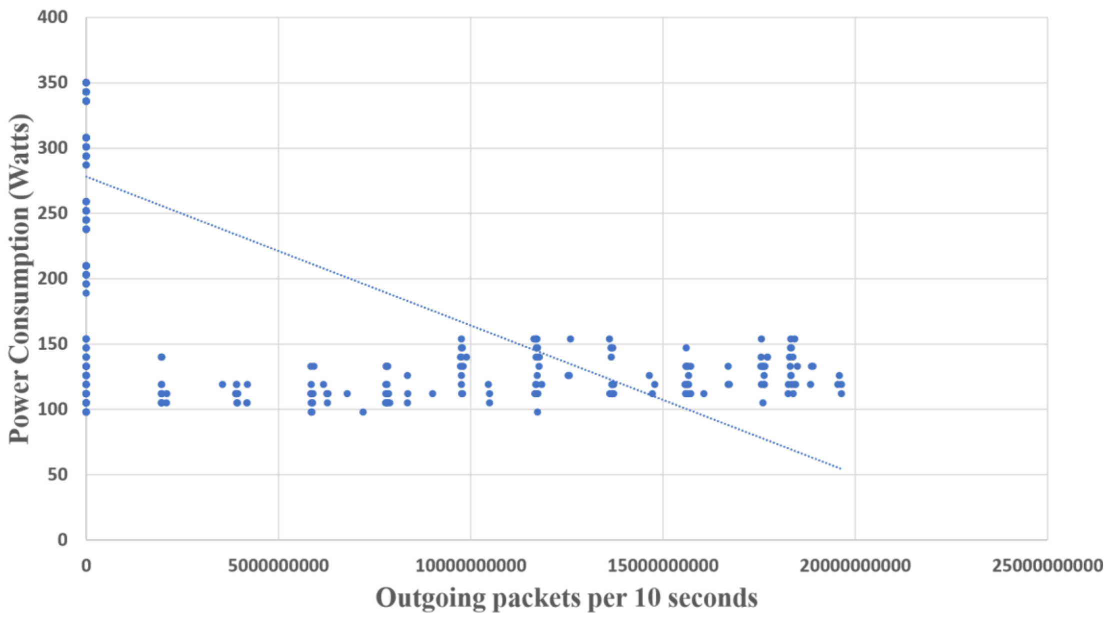


Fig. 27 Correlation diagram between power consumption and outgoing network traffic.

On the other hand, CPU cycles and dTLB-loads have positive correlation with the power consumption. Fig. 28 and Fig. 29 show that the intersection points of power consumption and CPU cycles as well as dTLB-loads are relatively close to a linear trendline. We calculated the R-squared value for each of these two features. In case of the CPU cycles the R-squared value is 0.8003, and for the dTLB-loads is 0.9672. The Rsquared values of the features that construct our prediction model, have a better fit to a linear regression line.

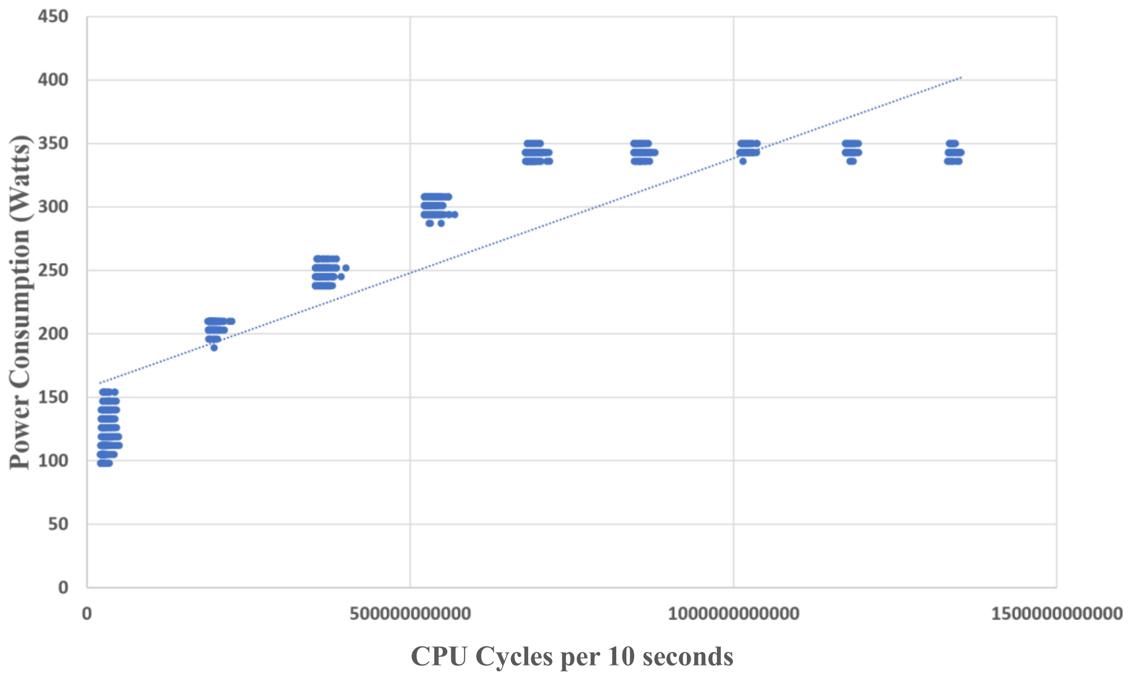


Fig. 28 Correlation diagram between power consumption and CPU cycles.

Further analysis on the behavior of data, for the workloads that our prediction model is less accurate, is considered as future direction of this work. Alternative models of supervised learning such as classification, could give more accurate predictions for independent workloads.

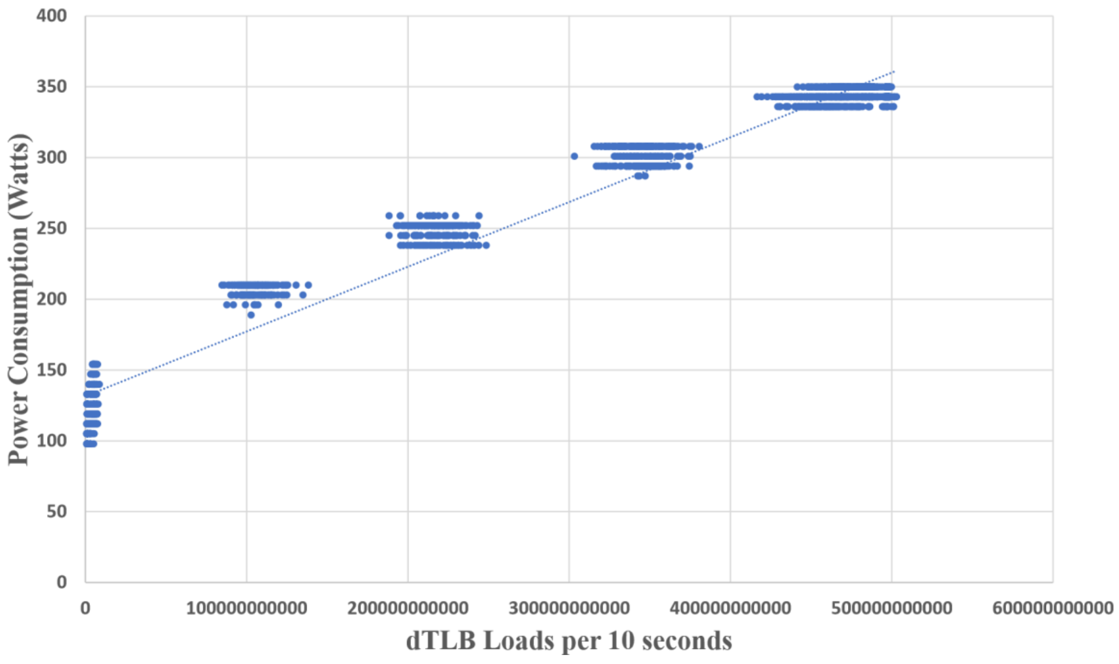


Fig. 29 Correlation diagram between power consumption and dTLB-loads.

Chapter 5

Conclusions

5.1. Summary

We created the power profile of the server. Leveraging machine learning models, we followed a fine-grained approach to derive the features that have the highest contribution to the power consumption. CPU cycles and dTLB-loads impact the energy consumption of the server the most. If we collect these two features from the processor of the server, we can estimate its power consumption. The power model can be used for predicting energy consumption of the server when these two regression variables are available. The advantage of the model is the fact that it predicts power consumption at system-level with average accuracy of 94.66% regardless of the server's workload, and is independent of individual server's components usage.

The power model is based on analysis of HPCs. An alternative direction, is a feasibility study of constructing a prediction model if the only available variables are CPU load, memory and Input/Output of the server. There are various questions regarding the accuracy of the power model against different types of hardware. We want to scale the model to predict the energy consumption of different types of server, e.g. Open Compute Project (OCP) type of server. If we can prove that our existing power model estimates the power consumption of other servers, it would be possible to compare their energy consumption for different workloads, and propose which server consumes less energy in different workload scenarios based on its usage.

Let us assume the following scenario; we have a MNO running network applications in a data center, using two different types of server, A and B. We use our power model to estimate the energy consumption for each server in various workload scenarios. We learn that server A consumes less energy as compared to B when running close to its maximum activity level. Knowing that server A is more energy efficient in high workloads, MNO can decide to consolidate its applications in more servers of type A and power off servers of type B; therefore, it can benefit energy consumption gains.