MEASURE ENERGY CONSUMPTION

PHASE 5

Abstract: The task of reducing the energy footprint of IT devices and software has been a challenge for Green IT research. Monitoring approaches have primarily focused on measuring the energy consumption of the hardware components of computing devices. The use of applications or software on our computer systems consumes energy and it also affects how various hardware components and system resources consume energy. Consequently, running web browsers applications will utilise considerable energy and battery consumption. In this research, we have run different types of experiments which involve the use of several measuring tools. Firsly, a joulemeter is used to monitor (and measure) the power consumed by the hardware and software while running web-based and stand-alone applications on several devices. Additionally, the tablet in-built battery status checker is used to measure the battery consumption when web-based applications are run on the device. 1 Introduction Green computing technology focuses on the efficient use of computing resources. In computing devices such as laptops, smartphones, tablets, or other mobile devices, energy consumption is the top priority because they are run on battery, with limited lifespan, as their source of power (Banerjee et al. 2007). With the increasing complexity of IT equipment, the energy consumption rate of these devices system also increases (Silven and Jyrkka, 2007). Most portable mobile device users are conscious of the energy usage by these devices and consequently, they look for ways through which the lifespan of the battery can be extended to serve them longer (Rahmati et al. 2007). Experiments relating to energy measurement could be at various levels: the hardware level; energy efficiency directive level (Simunic, et al. 2000); operating system (Sagahyroon, 2006); software application or data and user levels (Ravi, et al. 2008). Energy conservation is made possible through the use of different techniques which estimate or forecast energy consumption at the device and application level (Krintz, et al. 2004). The goal of green computing technology is to reduce carbon emission, maximize performance and prolong the lifespan of the computing resources. 1.1 Aim The aim of this paper is to discuss the results for several investigations conducted on the energy (and battery) consumption for running web-based and standalone applications on Windows and IOS portable computing devices. The following objectives will help to achieve this aim: Research Smart2020 report (The Climate Group and GeSI, 2008) predicts an increasing trend of BAU CO2 emissions for the ICT industry. The emissions growth rate for three ICT categories (end-user devices, telecommunication and networks, and data centers) is expected to decrease from 6.1% 3.8%. By 2020, the ICT industry's footprint is expected to rise to 1.3 GtCO2e (equivalent to 2.3% of global emissions by 2020). The PC (e.g. desktops, laptops, etc.) footprint (due to its embodied and usage emissions) is the highest (60%) followed by printers (18%), peripherals (13%), smartphones (10%), and tablets (1%). It is estimated that the footprint of end-user devices will grow at 2.3 percent per year to reach 0.67 GtCO2e in 2020 and thus, energy efficiency improvements in these devices and their proper usage are essential for reducing their overall footprint. 2.1 Energy Consumption of software

Green and sustainable software is a software product that has the smallest possible economic, societal, ecological impact as well as impact on human beings (Ahmed, et al., 2014). This has led to the introduction of various programmes and initiatives that encourages energy efficient software such as green software engineering and Eco-design software (Kaliterre, n.d.). According to the Greenhouse Gas Protocol (2012), applications are executed with an OS. They affect the power consumption of a device due to data requests and processing. Managing energy requires accurate measurement of the energy available and consumed by a system. This involves monitoring or estimating the resource and energy consumption of hardware and software (Noureddine, et al., 2013). However, a device's power consumption is subjected to the type of application and the task being performed which is evident in our experimental results presented in Section 4 of this paper. In order to reduce the overall power consumption for a web-based or standalone task, it will be necessary to provide users with an insight of the power consumption of the different web-based browser applications (e.g. Google Chrome, Internet Explorer, Mozilla Firefox, Safari, etc…) and also the resource hungry nature of many applications such as movie player and games. 2.2 Energy Consumption of Media Players Modern technologies incorporate a number of power management features to reduce power waste. Dynamic Voltage and Frequency Scaling (DVFS) can enable the CPU speed to be dynamically varied based on the workload which leads to a reduced power consumption during periods of low utilization (Liu, et al., 2008). The energy-aware dynamic voltage scaling technique has been used to reduce energy consumption in portable media players (Yang & Song, 2009). This scheme showed a relationship between frame size and decoding time. These two cited work merely discuss how energy consumption can be reduced using various techniques, but have not measured the actual amount of energy being consumed by the application. However, the energy consumption of Windows Media Player has been measured using the EEcoMark v2 tool (EecoMark, 2011) but the empirical details of the measurement have not been explicitly discussed. Media playback application power consumption has been analysed by Sabharwal (2011) using windows event tracing. Event tracing does not seem to be an appropriate method for measuring energy consumption because the process itself may have impact on the results. A comparative analysis of energy consumption of media players has been conducted by Techradar (2010). The energy consumption is monitored by playing a DVD on Windows Media Player (WMP) and VLC Media Player. Their research results show that the VLC Media Player is more energy efficient than Windows Media Player. However, the cited work has not mentioned which tool has been used for measurement and additionally, the experiment procedures have not been explitcitly discussed.

2.3 Metrics, Measure and Tools for Energy Consumption 2.3.1 Metrics Generally, software does not directly consume energy. However, running the software involves the hardware which consumes energy. Therefore, the resource usage metric such as the CPU usage, memory and disk usage are used as the measuring criteria (Mahmoud & Ahmad, 2013). It is important to analyse the energy efficiency of software by observing the amount of resource utilized versus the useful work performed. To measure the power consumption of a system, the power consumed by individual PC components must be measured. Therefore a system wide resource utilization monitoring technique at the user level seems to be more appropriate. 2.3.2 Measure and Tools There is a wide range of methods for measuring the energy consumption of a computer system. Generally, the measurement of energy consumption is grouped into three categories hardware, software and power models (Noureddine, et al., 2013). Measuring energy consumption of hardware using devices such as wattsup1 and the method described by McIntire and colleagues (2007) yields a precise value. PowerScope is a tool that uses a multi-meter to measure the energy consumption of applications (Flinn & satyanarayanan, 1999). This method is more precise because it can determine the energy consumption of a specific process and even procedures within the process. However, these methods have some limitations. It can only monitor hardware devices, not flexible, requires additional hardware and the value may fluctuate due to electro-mechanical issues. It is also difficult to upgrade to a more newer and precise monitoring without replacing the entire hardware. Power models are used to calculate the energy consumption of hardware and software. Kansal and Zhao (2008) use a generic automated tool to profile the energy usage of various resources components used by an application. This method is either too generic or coarse-grained and it is platform dependent (Seo, et al., 2007). The model proposed by Lewis and colleagues (2012) is an integrated model for the calculation of a system's energy consumption. More promising approaches are software energy measurement using energy application profiler (Noureddine, et al., 2013). In their contribution, Varrol and Heiser (2010) use Openmoko Neo Freerunner to decompose the energy consumption of each resource of a system. PowerAPI is an Application Programming Interface (API) used for monitoring the real time energy consumption of applications at the granularity of system process (Bourdon, et al., 2013). PowerAPI can also be used to estimate the energy consumption of a running process for hardware resources e.g. CPU or for hard disk or for both and many more other resources (Noureddine, et al., 2013). Energy consumption estimation in PowerAPI distinguishes the energy consumption for hardware resources and software blocks of codes. pTop is a process-level power profiling tool which provides information on the power consumption of the running processes in joules (Do, et al., 2009). It gives the power consumption values for the CPU, computer memory, hard disk and the network interface for each process. The energy consumed by an application is the sum of energy consumed by individual resources in addition to energy consumed by the interaction of these processes (Noureddine, et al., 2013). The windows version also uses the windows API to perform the same task. Intel energy checker is a software development kit SDK with the capability to provide the Application Programming Interface (API) required to define, measure, and share energy efficiency data (Intel, 2010). The SDK is developed with the intention to facilitate energy efficiency analysis

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.

Global Footprint (TWhr)

US Footprint (TWhr)

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

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.

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

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),

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

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).

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

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

$\boldsymbol{x_{i,j}}$

Preprocessing Function $\boldsymbol{g_j(x_{i,j})}$ Description

Processor 28 HPC event rates:

$x_{i,j}$

$(j\epsilon[1..28])$

$= \{$

$c_{i,1}$

d

$, \mathit{for\ cpu} - \mathit{cycles}$

$c_{i,j}$

$c_{i,1}$

$\mathit{for\ other\ HPCs}$

(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:

$g_j(x_{i,j}) =$

$x_{i,j} - \mathit{mean}(x_{i,j}$

$)$

$\mathit{standard\ deviation}(x_{i,j}$

$)$

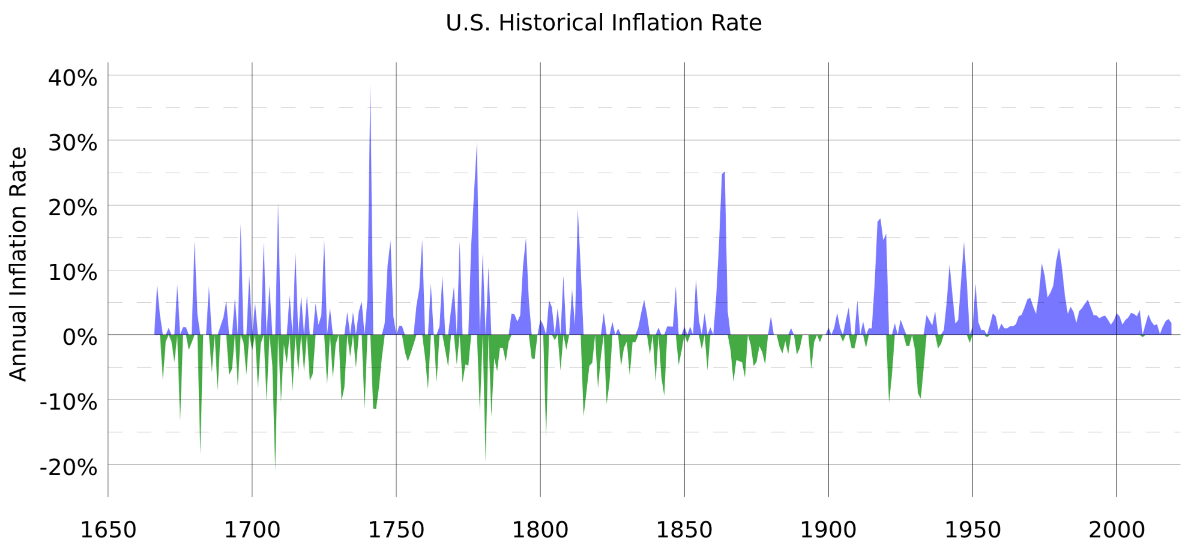
(5)

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, branch-
instructions, branch-
misses, bus-cycles,
cache-misses, cache-
references, instructions,
ref-cycles, L1-dcache-
load-misses, L1-dcache-
loads, L1-dcache-stores,
L1-icache-load-misses,
LLC-load-misses, LLC-

loads, LLC-store-
misses, LLC-stores,
branch-load-misses,
branch-loads, dTLB-
load-misses, dTLB-
loads, dTLB-store-
misses, dTLB-stores,
iTLB-load-misses,



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, cache-misses, cache-references, CPU cycles, instructions, ref-cycles, L1-dcache-loadcache-loads, L1-dcache-stores, L1-icache-load-misses, LLC-load-misses, LLC-loads, LLC-store-misses, LLC-stores, branch-load-misses, branch-loads, dTLB-load-misses, 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 in a pure CPU stress test and also well explain the powerconsumption.Variable R2σ% #Outhost_df_df.root_free_SQ 0.992 0.216 0.289 00.996 0.146 0.195 12host_df_df.root_used 0.992 0.216 0.289 00.996 0.146 0.195 12perf_instructions 0.991 0.232 0.310 00.996 0.143 0.191 12perf_cycles 0.991 0.233 0.310 00.997 0.144 0.192 12Table IBUR N CPU.Figure 2. Burn CPU residuals for variable host_df_df.root_free_SQ.Figure 2 shows the model residuals when using a modelwith only the variable host_df_df.root_free_SQ as singleexplanatory variable. There are only a few outliers, never-theless influencing σsignificantly.B. Memory LoopThe dataset "Mem Loop" consists of 2017 completedata records (without N/A values). Table II shows the bestvariables explaining the power,Variable R2σ% #Outperf_context.switches 0.998 0.268 0.339 00.999 0.213 0.270 18perf_cache.references 0.998 0.285 0.361 00.999 0.196 0.248 25perf_context.switches_SQ

0.977 0.919 1.163 0Table IIMEMORY LOOP.Figure 3. Loop memory residuals for variable perf_context.switches.Figure 3 shows the model residuals when using a modelwith only the variable perf_context.switches as single ex-planatory variable. Again a few outliers are visible, whichhowever do not significantly influence the results. Also,it is obvious that the residui do exhibit a deterministiccomponent. In fact this deterministic dependence could beexploited by further refining the regression model, e.g., byadding a sin()-like function. It turns out that this is notreally necessary since the one-variable model is already goodenough such that any model extension is superfluous.C. NetworkThe dataset "Network" consists of 1937 complete datarecords (without N/A values). Table II shows the bestvariables explaining the power,Variable R2σ% #Outhost_interface_if_octets.eth1_tx 0.954 0.178 0.247 00.983 0.106 0.147 13host_interface_if_packets.eth1_rx 0.953 0.181 0.251 00.980 0.117 0.163 13host_interface_if_packets.eth1_tx 0.951 0.183 0.255 00.981 0.114 0.159 13perf_cycles + 0.967 0.152 0.211 0host_interface_if_packets.eth1_tx 0.991 0.079 0.110 16perf_instructions + 0.967 0.152 0.211 0host_interface_if_packets.eth1_tx 0.991 0.079 0.110 16perf_cycles + 0.967 0.152 0.211 0host_interface_if_octets.eth1_tx 0.991 0.077 0.108 17Table IIINET WOR K.Figure 4. Network residuals for variable host_interface_if_octets.eth1_tx.Figure 4 shows the model residuals when using a modelwith only the variable host_interface_if_octets.eth1_tx assingle explanatory variable. This variable makes sense whenkeeping in mind that the workload mainly stresses thenetwork card. Furthermore, Table III and Figure 4 showthat although two variables are enough to explain the powerconsumption, there are a few outliers that do have a signifi-cant influence in the result. Removing only 13 out of 1937data records increases R2to 0.98 for one, and 0.99 for twovariables.D. Tar KernelThis dataset contains only 18 complete data records.Inasmuch it was not possible to reduce the number of
explanatory variables by using a full model first, and thus,searching for optimal variable combinations had to includeall 330 explanatory variables (K=330). This was partiallycompensated by the fact that regression for a small dataset ismuch faster. Nevertheless, the resulting run times were muchhigher. Table IV and Figure 5 show the regression results.It can be seen that mainly host oriented variables performbest, e.g., number of blocked processes, disk writing load,but also the load averages.Variable R2σ%host_processes_ps_state. 0.911 0.661 0.816blocked_valuehost_processes_ps_state. 0.911 0.660 0.816blocked_value_SQhost_disk.sda_disk_time_write_SQ 0.911 0.661 0.817host_cpu.2_cpu.wait_value_SQ + 0.984 0.290 0.358host_load_load_midterm_SQpid_kB_rd_s_SQ + 0.983 0.295 0.364host_load_load_midterm_SQhost_cpu.2_cpu.wait_value_SQ + 0.983 0.300 0.370host_load_load_shortterm_SQTable IVTAR KE RNE L.Figure 5. Tar kernel residuals for variables host_cpu.2_cpu.wait_value_SQ+ host_load_load_midterm_SQ.E. Disk ReadThis dataset contains 2018 complete data records. As canbe seen in Tables V and VI, and Figure 6 there are a fewoutliers that have a significant impact on the model accuracy.Removing them results in a sufficient fit though. We alsocomputed models for the optimal triple of variables, whichhowever did not improve the model quality significantly. Wethus omitted them in Table V. Interestingly single variablesdo not include disk related variables, but when sequentiallyadding variables, pid_kB_rd_s_SQ turns up at place third,so it is definitely related to the power consumption.Table VI shows that for six variables R2reaches up to0.98 without outliers, more variables though do not improvethe accuracy any more.F. Disk WriteThis dataset contains again

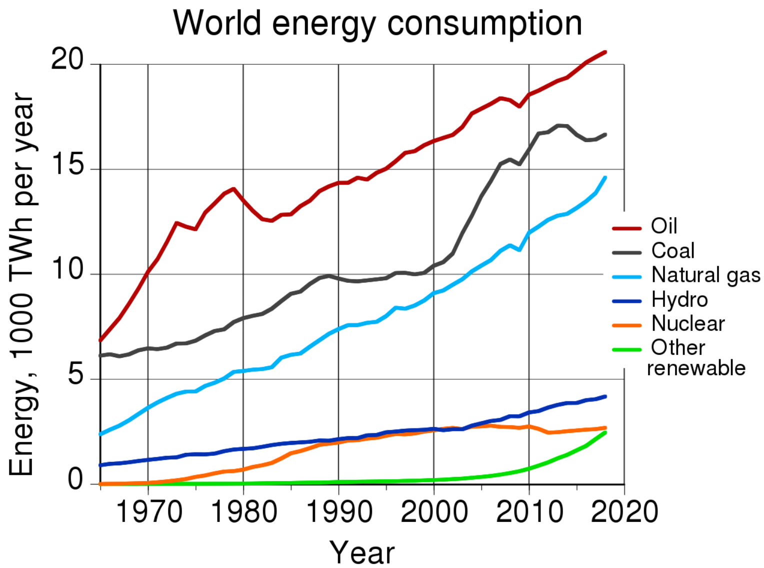
2018 non-empty data records.Similar to "Disk Read", this dataset also suffers from aVariable R2σ% #Outperf_cache.misses 0.824 0.698 0.964 00.956 0.337 0.465 18perf_context. 0.791 0.761 1.05 0switches_SQ 0.946 0.361 0.499 22perf_cache. 0.768 0.803 1.109 0references_SQ 0.963 0.298 0.412 23perf_cache.references + 0.941 0.405 0.559 0pid_system_SQ 0.972 0.277 0.382 12perf_cycles + 0.939 0.410 0.566 0pid_system_SQ 0.971 0.281 0.388 12perf_cache. 0.939 0.412 0.569 0references_SQ + 0.971 0.283 0.391 12pid_system_SQTable VDISK READ.Variable R2σ% #Outperf_cache.misses 0.824 0.698 0.964 00.956 0.337 0.465 18perf_cache.misses_SQ 0.909 0.502 0.694 00.956 0.337 0.465 17pid_kB_rd_s_SQ 0.919 0.474 0.655 00.970 0.280 0.387 18pid_system_SQ 0.937 0.417 0.576 00.971 0.274 0.379 19perf_cache.references_SQ 0.949 0.377 0.520 00.979 0.236 0.326 14perf_instructions_SQ 0.951 0.367 0.508 00.980 0.231 0.320 14Table VIDISK READ,SEQUENTIALLY ADDING SINGLE VARIABLES.handful of severe outliers. Figure 7 and Tables VII and VIIIshow the details. Here it can be seen that power for thisworkload can be best described without using disk relatedvariables. In general the models perform a little worse fordisk related workload (read and write), but when addingmore variables the model quality quickly rises significantly.G. All DatasetsThe previous sections show that our approach yields goodresults for individual types of work load. In this subsectionwe derive models for a dataset containing all previouslyanalyzed datasets. Since the datasets are quite heterogeneousit can be assumed that only 1,2 or 3 variables will not yieldgood models. However, Table IX shows that three variablesalready yield a good model. The main result however isshown in Tables X and XI, showing nine variables thathave been added sequentially. As can be seen, even withoutoutlier removal, these variables explain 99% of the totaldata variance, yielding them to be excellent predictors ofthe "All" dataset.The sequential model shown in Table X also shows thecoefficient of variable perf_cache.misses_SQ, which is in theorder of magnitude of 10−14. This is explained by the factthat the maximum of this variable (squared number of cachemisses per second) is also in the order of magnitude of 1014.

Figure 6. Disk read residuals for variables perf_cache.references +pid_system_SQ.Variable R2σ% #Outperf_cache.misses 0.768 0.588 0.822 00.925 0.297 0.415 19perf_context.switches_SQ 0.635 0.738 1.032 00.955 0.211 0.295 22perf_context.switches 0.628 0.745 1.0414 00.919 0.281 0.393 23perf_cycles_SQ + 0.895 0.396 0.553 0host_cpu.0_cpu. 0.967 0.216 0.303 12system_value_SQperf_instructions_SQ + 0.895 0.397 0.555 0host_cpu.0_cpu. 0.967 0.217 0.303 12system_value_SQperf_context. 0.893 0.400 0.560 0switches_SQ +host_cpu.0_cpu. 0.970 0.202 0.283 14system_value_SQperf_context.switches + 0.901 0.385 0.539 0perf_instructions_SQ + 0.969 0.211 0.295 10host_cpu.0_cpu.system_value_SQperf_context.switches + 0.901 0.385 0.539 0perf_cycles_SQ + 0.969 0.211 0.296 10host_cpu.0_cpu.system_value_SQperf_cache.misses + 0.900 0.386 0.539 0perf_instructions_SQ + 0.970 0.203 0.283 16host_cpu.0_cpu.system_value_SQTable VIIDISK WRITE.Figure 8 shows the model residuals when using ninevariables. Residuals are taken from the data sets in the order"Burn CPU", "Mem Loop", "Network", "Tar Kernel", "DiskRead" and "Disk Write". As can be seen residuals are notpurely random, but also depend on the data set, but not toa high degree. This is further investigated in Section IV-H.Figure 9 show how the models improve when addingmore and more variables sequentially, i.e., the ECDFs ofthe residuals when using 1, 3, 5, 7, or 9 variables addedsequentially. It can clearly be seen that the 9-variables modelis superior and yields

excellent results.Finally, Figure 10 shows the quantiles of the residualsplotted against the theoretical quantiles of normal distri-butions. The more straight lines the curves are, the moreVariable R2σ% #Outperf_cache.misses 0.761 0.588 0.822 00.925 0.297 0.415 19perf_cache.misses_SQ 0.816 0.524 0.732 00.919 0.308 0.430 20host_cpu. 0.865 0.449 0.627 00_cpu.system_value_SQ 0.932 0.296 0.414 21perf_context.switches_SQ 0.906 0.374 0.523 00.973 0.196 0.274 13Table VIIIDISK WRITE,SEQUENTIALLY ADDING SINGLE VARIABLES.Figure 7. Disk write residuals for variables perf_context.switches +perf_instructions_SQ + host_cpu.0_cpu.system_value_SQ.they follow a normal distribution. As can be seen, themodel using the optimal triple (O3), as well as the modelusing 9 variables show a large region between -3.5 and 2.5where they follow a normal distribution nicely. Outside thisregion, both models show differences in their tails causedby outliers.H. Global Model Applied to all DatasetsTable XI shows how a model with 9 variable explainsall data sets in total. The question now is how this globalVariable R2σ% #Outperf_cache.references_SQ 0.770 2.003 2.708 0perf_cache.misses_SQ 0.767 2.017 2.728 00.773 1.987 2.686 16perf_cache.references 0.683 2.353 3.181 0pid_usr + 0.920 1.181 1.597 0perf_cache.references_SQ 0.937 1.046 1.414 57perf_instructions_SQ + 0.917 1.207 1.632 0perf_cache.references_SQ 0.933 1.080 1.460 57perf_task.clock.msecs_SQ + 0.916 1.208 1.633 0perf_cache.references_SQ 0.933 1.078 1.457 57perf_cache.references + 0.963 0.807 1.091 0perf_instructions_SQ + 0.976 0.646 0.874 62pid_RSS_SQperf_cache.references + 0.963 0.807 1.091 0perf_instructions_SQ + 0.976 0.646 0.874 62pid_MEM_SQperf_cache.references + 0.963 0.807 1.091 0perf_instructions_SQ + 0.976 0.646 0.874 62pid_VSZ_SQTable IXALL DATAS ETS .

Variable Coeff Coeff ErrIntercept 7.056e+01 7.781e-03perf_cache.misses_SQ -8.195e-15 1.671e-15pid_usr 8.401e-02 1.879e-04host_irq_irq.26_value 6.882e-03 6.278e-05pid_system_SQ 3.717e-03 3.237e-05perf_context.switches 6.266e-04 1.107e-05pid_kB_wr_s 1.737e-04 4.696e-06host_disk.sda_disk_merged_write -5.270e-04 1.930e-05perf_cache.references 1.190e-06 1.357e-08pid_VSZ -1.601e-05 2.064e-07Table XALL DATAS ETS ,REGRESSION COEFFICIENTS AND ERRORS OF THECO EFFIC IEN TS . ALL C OE FFICI EN TS AR E SI GNI FIC ANT LY DIFF ERE NTFRO M ZER O (P-VALUE S<1.0E-7)Variable R2σ% #Outperf_cache.misses_SQ 0.767 2.017 2.728 00.773 1.987 2.686 16pid_usr 0.916 1.209 1.635 00.935 1.062 1.437 56host_irq_irq.26_value 0.950 0.934 1.263 00.971 0.706 0.955 63pid_system_SQ 0.966 0.772 1.043 00.977 0.634 0.857 48perf_context.switches 0.976 0.651 0.881 00.986 0.491 0.663 65pid_kB_wr_s 0.980 0.593 0.801 00.989 0.426 0.577 78host_disk. 0.983 0.541 0.732 0sda_disk_merged_write 0.989 0.430 0.581 92perf_cache.references 0.985 0.508 0.687 00.990 0.407 0.550 77pid_VSZ 0.991 0.402 0.544 00.995 0.282 0.382 90Table XIALL DATAS ETS ,SEQUENTIALLY ADDING SINGLE VARIABLES.Figure 8. All datasets residuals for the sequential model using 9 variablesshown in Table X.model performs when applied to each individual data set.Table XII shows the respective regression results. As canbe seen the global model performs outstandingly well, andexplains the power consumption for every single dataset. Theonly data set having some deviances is "Disk Write", butwhen removing only 12 outliers, the situation dramaticallychanges, and 97% of the variance is explained, which is anexcellent result.Figure 9. All datasets, ECDFs of the residuals, when using 1, 3, 5, 7, or9 variables added sequentially. O3 denotes the optimal combination of 3variables.Figure 10. All datasets, Q-Q-plot of the residuals, when using 1, 3, 5, 7,or 9 variables added sequentially. O3 denotes the optimal combination of3 variables.V. POWE R CONSUMPTION OF A SINGLE PROC ES SThe above derived models describe the power consump-tion of a computer by a combination of system wide vari-ables Yi, i = 1, . . . , I, as well as variables Xjl, j = 1, . . . , Jdescribing individual process pl, l = 1, . . . , L. Let the powerconsumption of a computer with no load be denoted byP0(the intercept), and the respective coefficients of theregression model be called αifor system wide variables, andβjfor per process variables. The linear model then relatesData set R2σ% #Out"Burn CPU" 0.992 0.225 0.3 0"Mem Loop" 0.999 0.144 0.24 0"Network" 0.944 0.196 0.27 00.974 0.134 0.19 17"Tar Kernel" 0.965 0.556 0.69 0"Disk Read" 0.946 0.389 0.54 00.974 0.265 0.36 11"Disk Write" 0.89 0.407 0.57 00.965 0.221 0.31 12Table XIIAPP LYING T HE G LOB AL MO DE L FRO M TABLE X I TO T HE RE SPE CT IVEDATA SET S.

the power consumption Pof a computer in the followingway:P=P0+IXi=1αiYi+JXj=1βjLXl=1Xjl .(1)By settingSj=LXl=1Xjl ,(1) can be written asP=P0+IXi=1αiYi+JXj=1βjSj.(2)In order to derive the power caused by an individual processPl, we assume that the relevant system wide variablesare caused by the processes, which are themselves de-scribed by their individual per process variables. To verifythis assumption we explained the system wide variableshost_irq_irq.26_value and host_disk.sda_disk_merged_writeby the per process variables. The resulting R2were 0.956and 0.99, i.e., they can be almost entirely explained by perprocess variables. We therefore writeYi=JXj=1γij Sj,(3)and adapt (1) to a new formP=P0+IXi=1αiJXj=1γij Sj+JXj=1βjSj=P0+JXj=1Sj βj+IXi=1αiγij !=P0+JXj=1ηjSj(4)by definingηj=βj+IXi=1αiγij .Thus, we now describe the global power consumption byper process variables only. At this point we can derive thepower Plas caused by plto bePl=JXj=1ηjXjl .(5)It can easily be seen thatP=P0+LXl=1Pl,as it should be.VI. CONCLUSIONWe defined and implemented a measurement platform atthe University of Vienna which we used to measure thedependence of the power consumption of a standard PC ontosynthetic workload. By using multivariate regression weexplain this power consumption by a subset of 165 processand system variables we measured during our experiments.We explore this dependence in depth and demonstrate thatthe obtained model quality is very good. We derive a globalmodel for all data sets that also explains each single dataset excellently. Using simple linear analysis we also definethe power consumption of a single process.Although we present a multitude of results, the mainresult of this paper is actually the methodology itself. Thedescribed regression results merely demonstrate that ourapproach does yield excellent prediction results.Future works include choosing the best values to measurein order to reduce the impact of measurement on the system.ACKNOWLEDGMENTThis work was done during a short term scientific mission(Cost action 0804) of Georges Da Costa (IRIT, Toulouse,France) to the University of Vienna (Austria) from 8, Januaryto 12, February 2010, and was



EuropeanCommission.REFERENCES[1] R. Joseph and M. Martonosi, “Run-time power estimationin high performance microprocessors,” in ISLPED ’01: Pro-ceedings of the 2001 international symposium on Low powerelectronics and design. ACM, 2001.[2] I. Kadayif, T. Chinoda, M. Kandemir, N. Vijaykirsnan, M. J.Irwin, and A. Sivasubramaniam, “vec: virtual energy counters,”in PASTE ’01: Proceedings of the 2001 ACM SIGPLAN-SIGSOFT workshop on Program analysis for software toolsand engineering. ACM, 2001.[3] X. Ma, M. Dong, L. Zhong, and Z. Deng, “Statistical powerconsumption analysis and modeling for gpu-based computing,”in SOSP Workshop on Power Aware Computing and Systems(HotPower ’09), 2009.[4] S. Rivoire, P. Ranganathan, and C. Kozyrakis, “A comparisonof high-level full-system power models.” in HotPower, F. Zhao,Ed. USENIX Association, 2008.[5] M. J. Crawley, Statistics: An Introduction using R.Wiley, 2005, iSBN 0-470-02297-3. [Online]. linear model using PMCs and processors' temperature-along with their respective square root and logarithm-for real HPC workloads. The variables are selected according to their correlation with the power-a given variable is added to the model if its inclusion on the linear model, after calibration, has a better correlation than before. ...

Effectiveness of Neural Networks for Power Modeling for Cloud and HPC: It’s Worth It!

Article

May 20

Jean-Marc PiersonLeandro Fontoura Cupertino

View

Show abstract

... Based on these results, the authors proposed a model based on S_PMC as in Equation 12. In the same year, [34] conducted experiments to study the energy consumed by different applications. The authors used synthetic workload to stress the server and simultaneously measured the power consumption and 165 server performance indexes. ...

... However, the servers for model development, workload to stress the servers, and the power measurement technique used by these works are different. For instance, [9,20,33,34,38,61,64,75,79,80,90,104] used a single server for power model development, while [26,27,39,41,62,67,74,76,119,123] used multiple heterogeneous servers. Moreover, [34,79,80,90] used synthetic workload to stress the server, while [9, 20, 26, 27, 33, 38, 39, 41, 62, 64, 67, 74-76, 104, 119, 123] used different benchmarking applications and real-world workload traces. ...
... For instance, [9,20,33,34,38,61,64,75,79,80,90,104] used a single server for power model development, while [26,27,39,41,62,67,74,76,119,123] used multiple heterogeneous servers. Moreover, [34,79,80,90] used synthetic workload to stress the server, while [9, 20, 26, 27, 33, 38, 39, 41, 62, 64, 67, 74-76, 104, 119, 123] used different benchmarking applications and real-world workload traces. In addition, [76] used DW-6090 power meter, [90] used a power analyzer, [123] used Chroma 66202 power meter, [67] used IBM active energy manager, [61] used Voltcraft Energy Logger 4000, [26,39,75] used a AC power meter, [9] used Yokogawa WT210 power meter, [20,64] used WattsUp PRO ES power meter, [34] used a watt meter, [79] used a smart power meter, [62,119] used home-brew power meter, and [33] used a plogg power meter to measure the power consumption of the server. ...
Computing server power modeling in a data center: survey,taxonomy and performance evaluation
Preprint
Full-text available
Apr 2020

Leila Ismail

Huned Materwala
View
Show abstract
... Kansal 2 Wireless Communications and Mobile Computing CPU utilization, the number of missing last-level caches, and the number of bytes that were read and written, as discussed in [11]. Costa and Hlavacs proposed a generalized power consumption model of the server based on its component performance counters, such as CPU cycles per second, cache per second, and cache misses per second [12]. Beloglazov et al. presented a nonlinear model of servers that performs better than regression models [13]. ...
... Equation (11) is expressed to meet the reliability of the application layer user service F i , and the number of virtual machines to be deployed on the server is I i . Equations (12) and (13) indicate the resource constraints that need to be met to avoid server overload. Equation (14) means that to avoid server failure, resulting in unreliable user services, the number of virtual machines corresponding to the deployment of the same service by up to the user is 1 on the same server. ...
Energy Consumption Optimization of an IoT Monitoring Center Based on a Max-Min Ant Colony Algorithm
Article
Full-text available

Feb 2023WIREL COMMUN MOB COM
Yinghua Tong
View
Show abstract
... The term N LLCM represents the number of the missing LLC during T , and α mem and γ mem are the linear model parameters. A more generalized power consumption model is presented in [88] based on the server's components performance counters (i.e., CPU cycles per second, references to the cache per second, cache misses per second), as given in (11). Later, the power consumption model in (11) is further extended by Witkowski et al. [89] by including the CPU temperature in the model. ...
A Review of Data Centers Energy Consumption And Reliability Modeling
Article
Full-text available
Nov 2021

Kazi Main Uddin Ahmed

Math Bollen

Manuel Alvarez
View
Show abstract
... One can do it soon after procuring h k in the Cloud data center. The value of memory access rate depends on the number of last level cache misses, and it can be obtained from the hardware performance counters at a low overhead [23,33]. Figure 1 shows the model of the Task Scheduler. ...
Energy and Makespan Aware Scheduling of Deadline Sensitive Tasks in the Cloud Environment
Article
Full-text available
Jun 2021

Anurina Tarafdar

Mukta Debnath

Sunirmal Khatua

Rajib K Das
View
Show abstract
... Da Costa et al. [DCH10] evaluate the power consumption of a PC by using performance counters, then extend the conception to predict the power consumption of single applications. Training data is collected by running several applications and synthetic benchmarks. ...

Evaluating and modeling the energy impacts of data centers, in terms of hardware / software architecture and associated environment
Thesis
Mar 2020

Yewan Wang
View
Show abstract
... Based on these results, the authors proposed a model based on S_PMC as in Equation (12). In the same year, Reference [34] conducted experiments to study the energy consumed by different applications. The authors used synthetic workload to stress the server and simultaneously measured the power consumption and 165 server performance indexes. ...
Computing Server Power Modeling in a Data Center: Survey, Taxonomy, and Performance Evaluation
Article