FRESCO: A Framework to Estimate the Energy Consumption of Computers

Pavel Efros, Erik Buchmann and Klemens Böhm
Karlsruhe Institute of Technology (KIT)
76131 Karlsruhe, Germany

***Abstract*—Many application areas, e.g., demand response, energy accounting or energy-aware scheduling, require estimates of the energy consumption of computer systems. However, existing estimation approaches often make restrictive assumptions regarding the effort at setup time or run time that is acceptable, they are tailored for specific hardware or software, or they cannot provide accuracy guarantees for the estimates. In this paper, we introduce *FRESCO*, a FRamework for the Energy eStimation of COmputers. FRESCO is a flexible framework for the estimation of the energy consumption of a wide range of computer systems. In particular, FRESCO considers technical information of the hardware manufacturer, system specifications like the average load, information that have been sampled at run time, e.g., the time the CPU spent in a specific state, and energy consumption profiles that might have been learned at setup time. Based on accuracy requirements and information available, FRESCO deploys and executes appropriate estimators. We have evaluated FRESCO with three real-world use cases. Our evaluation shows that FRESCO produces meaningful estimates for a wide range of analytical scenarios.**

I. Introduction

Vereecken et al. [1] has estimated the share of energy consumed by computers to be 7.15% of the total electricity consumption, and it is estimated to grow further (approx. 14.6% by 2020). Thus, the energy consumption of IT systems is an important cost driver for any enterprise, and quantifying this type of consumption reliably is a cornerstone for many current business models. For example, the energy consumption has a significant impact on the total costs of ownership of a data center. It must be considered for total absorption accounting. Furthermore, in the context of the smart grid, that data gives way to new business models, e.g., by scheduling data centers according to an oversupply of renewable energies.

One way to quantify this type of energy consumption and integrate it into the *Smart Grid* is to deploy a smart meter for each computer system [2], [3]. Since it is expensive to equip existing hardware components with such meters, this is doable only in rare cases. Recent research has provided methods to *estimate* the energy consumption of computer hardware, e.g., based on nameplate information [4], sophisticated hardware models [5], or by profiling system and component power usage [6], [7]. However, all of these approaches have different characteristics in terms of setup effort, estimation effort, estimation accuracy and hardware requirements. It is difficult to decide when to use which approach and how to determine meaningful estimation parameters, as well as the accuracy requirements of a given application.

Our goal is to devise a framework for the estimation of the energy consumption of computer systems that considers many different accuracy and effort measures. This is challenging, because today's computer systems come with a wide range of different usage parameters and technical specifications, and we have to consider use cases that differ very much in the accuracy required and the effort acceptable for the operator.

In this paper we introduce *FRESCO*, a FRamework for the Energy eStimation of COmputers. FRESCO consists of a configurable set of estimators, and a workflow to set up and run an instance of an estimator. In particular, FRESCO is able to (a) suggest a set of appropriate estimators for computer energy consumption according to the effort the operator is willing to invest and to the requirements of a certain application, and (b) to execute an instance of the selected estimator with settings that are appropriate for the application. Depending on the kind of estimator, FRESCO can estimate the energy consumption of a computer from various parameters. Such parameters include (1) hardware characteristics, e.g., the energy consumption of a hard disk as specified by its vendor, (2) usage information like CPU load and network activity, and (3) calibration data, e.g., an energy consumption profile that has been recorded by an energy meter for a specific hardware.

FRESCO explicitly models the trade-off between the accuracy of the estimation and the effort of obtaining technical specifications, building energy profiles or measuring CPU usage information. For example, FRESCO can estimate the energy consumption of a PC with a high precision in hourly intervals, but also with a lower precision in intervals of a few seconds. Furthermore, FRESCO can provide upper and lower bounds for the estimation, and it considers heterogeneous hardware components and heterogeneous loads. In this paper, we make the following contributions:

- We introduce three flexible power-estimation models that generalize state-of-the-art approaches to cover a wide range of accuracy requirements and computer systems.
- We describe FRESCO, which integrates these models into an estimation workflow that allows to choose, configure and run an estimator depending on the use case.
- We evaluate three use cases, namely data center management, demand response and energy accounting.

Our evaluation confirms that FRESCO can estimate the energy consumption of IT systems by considering a wide range of parameters and with an accuracy of up to 95%, and that it supports many different use cases.

Paper structure: Sections II and III describe three applications for energy estimation and two respective effort classes. Section IV introduces FRESCO, which Section V evaluates. Section VI reviews related work, and Section VII concludes.

II. Application Scenarios

In this section, we describe three different use cases that cover the spectrum of energy-aware applications for FRESCO.

1) Energy-Aware Management of Data Centers: Increasing the performance per watt is a key performance optimization for data centers [8], [9]. For this purpose, it is important to obtain the energy consumption of a complex IT system as early as the design time of the data center or the allocation time of the various computing workloads. Recent approaches, e.g., in the area of energy-aware cloud data centers [10] or energy management for warehouse-sized computing centers [11], distinguish (1) the (static) energy consumption at idle state, and (2) the (dynamic) energy consumption depending on the workload of the target system. This is important to design the power distribution infrastructure, to decide about computing hardware acquisitions or to find out if a scheduled workload exceeds the cooling capacity.

Thus, two different accuracy requirements exist: It must be possible (a) to provide estimates for the typical case that are sufficiently accurate to make educated decisions for hardware acquisitions, and (b) to provide upper bounds for the energy consumption in extreme cases. Both requirements must be fulfilled at design time or at allocation time, i.e., before the operator can measure the workload or the energy consumption. Furthermore, an estimator must consider that some in-depth hardware specifications might be unavailable at design time.

2) Demand-Response: Demand Response (DR) contains measures that influence energy-consumption patterns. For example, DR might be used to shift energy-intensive computing tasks to times of an energy surplus [12]. DR can be divided into (a) incentive-based DR and (b) time-based rates DR [12]. Incentive-based DR measures shift the energy consumption by providing, say, tariffs that reward to shift energy consumption into off-peak hours. In contrast, time-based rates DR makes use of static schedules.

Since a data center is a large, adjustable energy sink, it is particularly well suited to perform demand response measures [13]. To realize DR in a data center, an estimator must deliver continuous estimates of the energy consumption of the various IT components at run time. In particular, the estimates must be adequate to identify system states that produce energy-consumption peaks. Furthermore, the personnel costs and computational effort of the estimation must not exceed potential savings from DR, otherwise the use of DR is not justified. Finally, the estimator must cope with technical parameters on different levels of detail. For example, a coarse estimate could measure the average CPU load only, while a fine-grained approach might also consider the voltage and frequency of the CPU and the states of other hardware components.

3) Computer Energy Accounting and Billing: Energy accounting and billing of the IT infrastructure becomes more and more important. For example, in the context of total absorption accounting, an enterprise might wish to assign each benefactor (a good or a service) the energy costs required for its production [14].

Typically, computer Energy Accounting requires estimates of the consumption with a frequency of 15 minutes to one hour. Furthermore, the estimator must provide stochastic accuracy guarantees (e.g., an accuracy of $\pm$ 10%), that allows to assign the energy consumption of an IT system to a department or a product line. As for the previous scenarios, the estimator has to be applicable to a large variety of computer systems, with an effort that is adaptable.

III. The Classes of Effort

We have identified two classes of effort, which our framework must take into account.

The **Setup Effort** is necessary to set the estimator up and running. This includes collecting technical specifications of the energy consumption of certain hardware components, e.g., the energy consumption of the CPU in activity states like idle or sleeping. Furthermore, it contains the effort of installing a monitoring application to measure run-time parameters of the hardware usage, e.g., disk activity. Finally, the setup effort includes the calibration of an energy consumption profile for a given hardware, e.g., measuring the energy consumption while executing a benchmark application.

The **Run-Time Effort** includes the network overhead and the computational overhead of the estimation process, and the overhead of a monitoring application collecting hardware parameters like CPU frequency or rotation speed of the hard disks, if required by the estimator. The more parameters the estimator samples, the higher are the data volume transferred, the computational overhead and the complexity of the estimation.

We expect that the two effort classes will be traded for each other. For example, the same accuracy requirement can be met either (a) by measuring an energy consumption profile at setup time or (b) by using detailed specifications and usage parameters at run time.

IV. FRESCO

In this section we describe the workflow and the estimators of our FRamework for the Energy eStimation of COmputers.

A. The FRESCO Workflow

With "*Target System*" we refer to the computer system whose energy consumption FRESCO must estimate. "The *Operator*" is responsible for installing and maintaining the estimator on the target system. FRESCO consists of three subsequent stages "*Setup*", "*Configuration*" and "*Estimation*", as shown in Figure 1, which we explain in the following.

a) Setup: At the first stage, the operator quantifies the trade-off between effort and estimation accuracy for the target system. In particular, the operator specifies the categories of information obtainable from the target system. This includes:

- The nameplate information available, e.g., if the energy consumption of the network card can be obtained.
- The parameters measurable on the target system, e.g., CPU frequency, CPU voltage or hard disk activity.
- If it is possible to measure a consumption profile for the target system, and with which accuracy.

The type of estimates and their error guarantees influence the choice of the estimators FRESCO suggests.

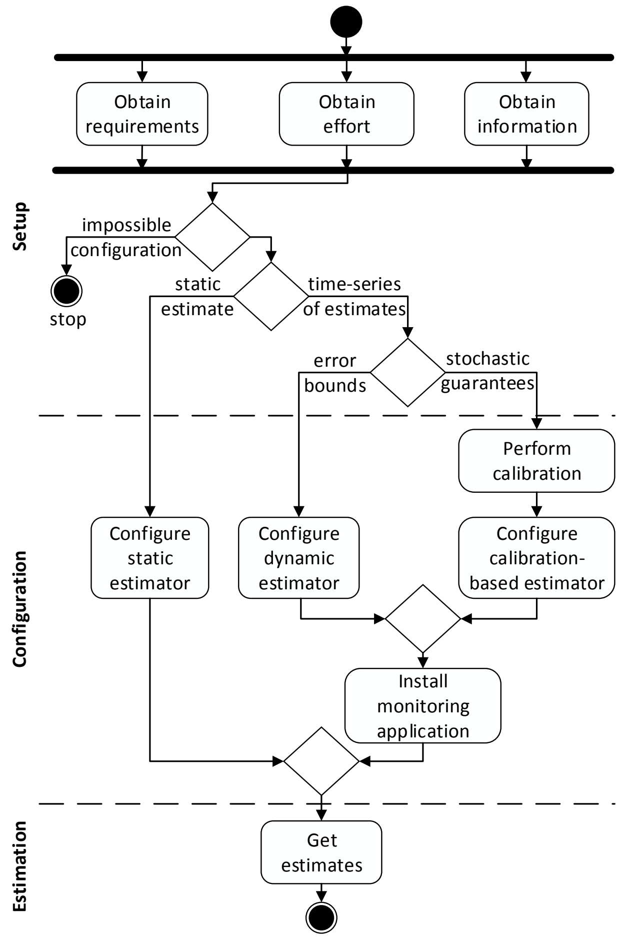

Fig. 1: FRESCO Workflow

At the end of the setup stage, FRESCO either indicates the operator that, given his input, estimation is impossible or lets the operator choose one or a combination of estimators. The former case happens, e.g., when the operator requires estimates with stochastic guarantees while calibration is not possible. In the latter case, FRESCO provides information on the estimation accuracy possible and the effort required for each estimator.

b) Configuration: At this stage, FRESCO helps the operator to configure the estimators selected, according to:

- The usage parameters that must be measured to meet the accuracy specified in the setup stage.
- The frequency at which the parameters must be measured.
- The energy consumption profile, if necessary.

If the operator has chosen a calibration-based estimator, FRESCO provides a set of benchmarks and guides the operator through measuring the energy consumption. The result of this stage is the combination of configured estimators.

c) Estimation: Finally, FRESCO runs instances of the chosen estimators with the configuration parameters just fixed on the target system and estimates its energy consumption.

B. The FRESCO Estimators

FRESCO can use static, dynamic or calibration-based estimators or a combination of them. In the following, we briefly sketch each estimator, and we discuss the effort required and their accuracy. More details can be found in a complementary technical report [15].

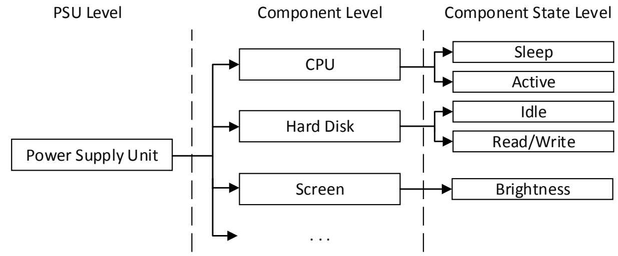

Fig. 2: Classes of Information Used by the Static Estimator

1) Static Estimator: Our static estimator is inspired by [4]. The approach in [4] is tailored to servers running a single application, similarly to the TPC (Transaction Processing Performance Council) benchmark suite [16], at peak load. Thus, it aggregates the peak power consumption of all hardware components. Since we are interested in estimates for a wide range of target systems operating at different loads, we have extended this approach. In particular, we have modeled a wide range of computer architectures and hardware components, e.g., laptop screens or motherboards of standard PCs. Furthermore, we consider three levels of detail to model the computer architecture, namely the PSU level, the component level and the component state level, as shown in Figure 2. We have obtained these levels of detail by abstracting from the parameter sets used in [4]. For the component-state level we consider all power-consumption states besides peak consumption.

The *PSU level* considers only the specification of the Power Supply Unit (PSU) powering the target system. In this case, the estimator calculates the total energy consumption E by using the maximum power PSU_{max} the PSU is able to supply and the run time ΔT of the target system:

$$E = \Delta T \cdot PSU_{max} \tag{1}$$

We use PSUmax, the only value available at this level of detail. If a higher accuracy is required, if more information is available, and if more effort at setup time is acceptable, FRESCO considers information at the *component level*. In this case, the consumption E is the sum of the power consumptions P_i of each component $i \in C$, $C = \{CPU, RAM, Hard\ Disk, ...\}$, multiplied with the run time ΔT:

$$E = \Delta T \cdot \sum_{i \in C} P_i \tag{2}$$

Note that we represent the CPU of a multicore system as a set of components. One component constitutes core-independent consumption, e.g., due to caches shared by all cores. The other components represent individual cores.

The *component state level* includes information on (1) the power consumption of the different states of the components of the target system and (2) the time the components typically spent in certain states. In this case, FRESCO obtains the total consumption E by summing up the energy consumption P_{ij} of each component $i \in C$ in a particular state $j \in S_i$ (the set of states of component i), multiplied with the time ΔT_{ij} each component typically is in this state:

$$E = \sum_{i \in C} \sum_{j \in S_i} \Delta T_{ij} \cdot P_{ij} \tag{3}$$

In the case of a virtualized environment, the static estimator incorporates virtual machines at the component state level, i.e., the operator maps components to virtual machines.

Component	Model	Power Consumption
CPU	Intel i5-3320M	Idle – 2.9 W Minimum active – 7.5 W Thermal Design Power – 35 W Maximum active – 80.56 W
Memory	Micron Technology 2x4 GB DDR3L SDRAM 800 MHz	Minimum – 0.3 W Typical – 1.48 W Maximum – 1.68 W
Hard Disk	Hitachi HTS725050 500 GB at 7200rpm	Sleep – 0.1 W Standby – 0.2 W Active idle – 1.0 W Read/write – 1.8 W

TABLE I: Laptop Components

The accuracy of a static estimator depends on the detail level used and on confidence intervals on the input parameters. Due to the lack of space, we briefly discuss only how to obtain bounds on the component state level: Each hardware manufacturer provides detailed data sheets containing the minimal, typical and maximal energy consumption of any hardware component. This information can be used to obtain upper and lower bounds on the energy consumption by using Equation 3.

Example 1: Consider a laptop as described in Table I. The lower bound on the consumption is the sum of the minima of each component, i.e., 2.9 W + 0.3 W + 0.1 W = 3.3 W.

Summary: *A static estimator might be sufficient for any application that does not need time series of estimates. It requires a small effort at setup time for obtaining the hardware specifications, and no effort at run time. The accuracy of this estimator depends on the detail level of its input values and the availability of tolerance bounds. In particular, the static estimator can provide bounds on the energy consumption.*

2) Dynamic Estimator: Our dynamic estimator models the energy consumption similarly to the static estimator, but installs a monitoring application on the target system to periodically measure detailed load information in real-time, e.g., CPU load or sleep times of the hard disk. Thus, our dynamic estimator generates time series of energy consumption data. Our dynamic estimator uses a monitoring application to record at run time in which state $j \in S_i$ the component $i \in C$ operates at time t. The energy consumption E_t at time t is the sum of the consumptions P_{it} of the components i:

$$E_t = \sum_{i \in C} P_{it} \tag{4}$$

The consumption E for a time interval $[t_p; t_q]$ is the sum of the consumptions at each point in time, multiplied with the period of time Δt between taking two consecutive samples:

$$E = \sum_{i=t_p}^{t_q} E_i \cdot \Delta t \tag{5}$$

The energy consumption P of the components can be modeled in different ways. For example, consider the consumption P_t^{CPU} of the CPU. Suppose that the monitoring application measures the state information "CPU load" l_t^{CPU}, and the operator knows the minimum and maximum power P_{min}^{CPU} and P_{max}^{CPU} the CPU can consume. In this case, P_t^{CPU} is:

$$P_t^{CPU} = P_{min}^{CPU} + l_t^{CPU} \cdot (P_{max}^{CPU} - P_{min}^{CPU}) \tag{6}$$

It is also possible to integrate specific models for multi-core systems [17] and to model virtual machines as components of the target system. While the accuracy of the static estimator depends on the knowledge about the typical load of the target system, our dynamic estimator samples such parameters. Thus, the accuracy of our dynamic estimator depends on the sampling frequency of the monitoring application. The reason is as follows: If a state changes between taking two consecutive samples, the estimator does not know to which extent the states were active. However, FRESCO provides upper and lower bounds on the energy consumption by assuming that a state change has taken place immediately before or after taking a sample. The upper (lower) bound is the maximum (minimum) of the power consumptions of the two consecutive samples.

Example 2: Suppose that the hard disk of a laptop (Table I) has been observed in standby at time t_3, and as idle at t_4. Thus, the lower bound on the consumption in interval $[t_3 : t_4]$ is $0.2W \cdot \Delta t$, and the upper bound is $1.0W \cdot \Delta t$.

Formally, consider a component with the sequence of states $S = (s_1, s_2, \cdots, s_n)$, ordered by the power P_i, the component consumes in state i. Let $s_t = (s_{t_1}, s_{t_2}, \dots)$ be the time series of the states of the component sampled at times $t_1, t_2, \dots$. The upper bound $E_{\Delta t}^u$ on the energy consumption during time interval Δt between consecutive samples t_j and t_{j+1} is:

$$E_{\Delta t}^u = \begin{cases} P_1 \cdot \Delta t \text{ if} & s_{t_{j+1}} = s_1 \wedge s_{t_j} = s_1 \\ \dots & \\ P_i \cdot \Delta t \text{ if} & s_{t_{j+1}} = s_i \wedge s_{t_j} \leq s_i \vee \\ & s_{t_j} = s_i \wedge s_{t_{j+1}} \leq s_i \\ \dots & \\ P_n \cdot \Delta t \text{ if} & s_{t_{j+1}} = s_n \wedge s_{t_j} \leq s_n \vee \\ & s_{t_j} = s_n \wedge s_{t_{j+1}} \leq s_n \end{cases}$$

We calculate the lower bound $E_{\Delta t}^l$ likewise.

Summary: *Our dynamic estimator is suitable for applications that require time series of the energy consumption of the target system at run time. Since this estimator also needs technical specifications, it requires a similar effort at setup time as a static estimator. The effort at run time depends on the number of parameters that the estimator samples, and on the sampling frequency. The accuracy of our dynamic estimator depends on the technical specifications and the sampling frequency. The estimator can compute bounds on the energy consumption.*

3) Calibration-Based Estimator: This estimator borrows from the Mantis approach [18], which estimates the power consumption of a system by correlating AC power measurements from a calibration phase with performance counters of the CPU. Our calibration-based estimator executes a detailed benchmark at setup time, which gradually stresses each system component in isolation. At the same time, a digital power meter records the actual energy consumption, and our monitoring application measures load information such as CPU frequency, hard disk usage, etc. FRESCO then builds a regression model.

More specifically, let $M^{CPU}(l, f), M^{Disk}(l), M^{RAM}(l)$ be the regression models obtained through calibration for the CPU, Hard Disk and RAM, and let l be the load of the component and f the frequency of the CPU. Given the load information l_t^{CPU}, l_t^{Disk}, l_t^{RAM} and f_t at time t, the energy E consumed in the time interval $[t_1, t_n]$ is:

$$E = \sum_{i=t_1}^{t_n} \left(M^{CPU}(l_i^{CPU}, f_i) + M^{RAM}(l_i^{RAM}) + M^{Disk}(l_i^{Disk}) \right) \tag{7}$$

The calibration-based estimator can derive stochastic accuracy guarantees from the regression model.

Summary: *The calibration-based estimator is well-suited for applications that require a calibrated zero point and stochastic guarantees on the estimation quality, such as billing. Due to the extensive calibration, this estimator comes with a very high effort at setup time. The calibration can be done automatically. Its effort is necessary only once during setup and depends on the number of components calibrated and their type. As an example, the calibration of the CPU of the laptop computer (Table I) took around 25 minutes. This particular CPU can operate at 16 different frequencies. At run time, the effort depends on the number of parameters that must be sampled and on the sampling frequency.*

Summing up everything, FRESCO can model a wide range of computer architectures and environments, and it can generate static ex-ante estimates as well as time series of estimates with a configurable estimation frequency and different accuracy guarantees. Furthermore, FRESCO considers different kinds of effort at setup time and run time. More details about our estimators can be found in [15].

V. EVALUATION

Our framework operates as intended if its estimates are appropriate for a wide range of applications. That is, FRESCO must let the operator decide on a tradeoff between accuracy and effort, according to the requirements of the application. Thus, we evaluate FRESCO by means of three use cases, and we measure the accuracy obtainable with a certain effort.

A. Measures

To evaluate how well FRESCO can estimate the real data measured by our digital multimeter, we have computed two metrics. The **Mean Absolute Percentage Error (MAPE)** measures the average of the percentual deviation between the actual values and the estimates:

$$MAPE = \frac{1}{n} \sum_{i=1}^{n} \frac{|y_i - y'_i|}{y_i}$$

where y_i are the actual values and y'_i are the estimates. n is the number of records in the dataset. MAPE of zero means that the estimated values perfectly match the ones measured. Correspondingly, the **Maximum Absolute Percentage Error (MaxAPE)** measures the maximum percentual deviation between the actual values and the estimates:

$$MaxAPE = \max_{i=1}^{n} \left(\frac{|y_i - y'_i|}{y_i} \right)$$

B. Evaluation Setup

We have evaluated three different datasets. The **Server Dataset** is about a mail server executing SpamAssassin. Its workload is a daily pattern with a low usage during the night and a high usage in the morning and afternoon hours. Load peaks occur when the server checks bulks of e-mails sent to large mailing lists. Table II shows the hardware components of this system. P-states are power-performance states of the processor. We have used a digital multimeter Wattsup PRO [19] (accuracy: 1.5%) to measure the energy consumption at every minute as a reference. Furthermore, our monitoring application has logged CPU usage, CPU frequency and hard disk drive usage with a sampling frequency of one second. Our measurements cover a period of three weeks.

Component	Model	Power Consumption
CPU	2 x AMD Opteron 275	Maximum – 95.2 W P-State #1 – 90.3 W P-State #2 – 75.9 W Minimum P-State – 36.1 W Halt Mode – 16.6 W
Memory	Micron Technology 4x1 GB DDR400 PC3200	Minimum – 9.9 W Typical – 36.4 Maximum – 87.48 W
Hard Disk	2 x Seagate ST937401 2x74 GB at 10000 rpm	Maximum – 10.2 W Idle – 5.07 W Minimum – 4.69 W

TABLE II: Server Dataset

The **Desktop Dataset** contains three weeks of energy consumption, CPU usage and CPU frequency measured on an office computer (Table III) with a sampling frequency of one second. Its workload is the result of typical secretarial tasks, e.g., MS Office, Internet Explorer and administrative applications. The workload rarely reaches the maximal computing capacity, and the computer is active only during office hours.

Component	Model	Power Consumption
CPU	Intel Pentium Dual Core E5300	Deeper Sleep – 4 W Extended Halt – 8 W Thermal Design Power – 65 W Maximum – 92.9 W
Memory	Crucial Memory 2x2 GB DDR2 SDRAM 800 MHz	Minimum – 3.65 W Typical – 5.1 W Maximum – 10.4 W
Hard Disk	Western Digital WD2500AAJS 250 GB 7200 rpm	Standby – 0.73 W Sleep – 0.73 W Idle – 4.92 W Read/Write – 5.36 W

TABLE III: Desktop Dataset

For the **Laptop Dataset** we have measured the same parameters as for the desktop dataset, over a period of two weeks. The laptop (Table I) has been used for research purposes, i.e., the system load does not follow any regular pattern and shows idle periods as well as maximum load conditions.

C. Use Cases

We have evaluated FRESCO with the three use cases described in Section II.

1) Energy-Aware Data Center Management: Our first use case (cf. Subsection II-1) requires ex-ante estimates of the energy consumption depending on a predefined workload. The estimates must be sufficiently accurate for informed management decisions, e.g., it must be possible to find out if one target system requires significantly more energy for a certain workload than another one. Furthermore, it must be possible to find out if a certain workload might exceed the cooling capacity in the worst case. Thus, FRESCO proposes the static estimator model. To evaluate this scenario, we let FRESCO estimate upper and lower bounds on the energy consumption, and the average energy consumption for a typical workload.

a) Minimal and Maximal Consumption: We let FRESCO exemplarily estimate upper and lower bounds on the consumption of a server. With our use case, the operator specifies manufacturer information on the component level for CPU, RAM and disk, as specified in Table II. Our target system consumes the most energy when the CPU operates with the highest frequency at the highest voltage allowed in the specifications (95.2 W), and when the hard disk is in read/write mode at the highest rate of I/Os per second (10.2 W). The RAM consumes at most 87.48 W. Our server has two CPUs and two disks. Thus, the upper bound for the energy consumption is (2·95.2 + 2·10.2 + 87.48) W = 298.28 W. Correspondingly, the lower bound for the power consumption is 52.48 W. Our measurements confirm that these bounds apply. The bounds can be narrowed if the operator is able to specify a limit for the time each component is in a specific state (cf. Equation 3).

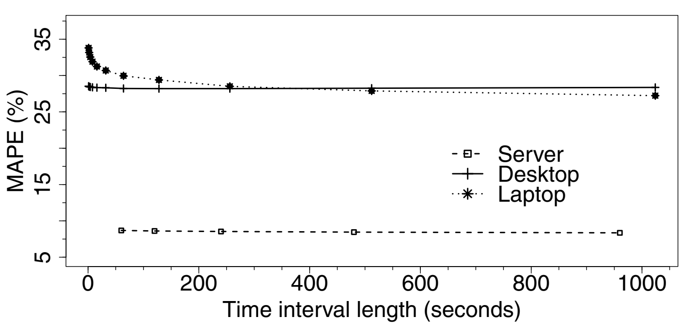

Fig. 3: MAPE, Static Estimator

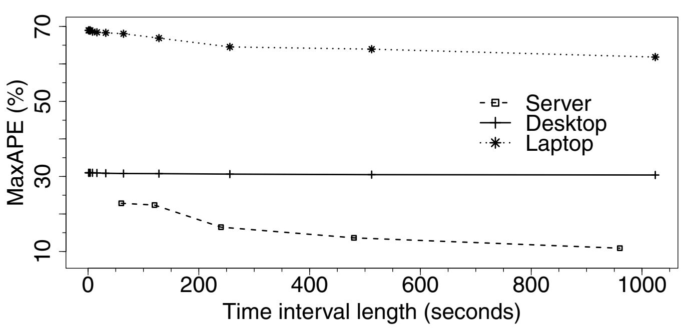

Fig. 4: MaxAPE, Static Estimator

b) Average Consumption: Now we assume that the operator wants FRESCO to estimate the energy consumption for an average system load of 50%, in order to assess the typical cooling requirement. To evaluate the accuracy of the estimates, we aggregate the energy consumption, which we have measured with a frequency of up to one second, to time intervals from one second to 16 minutes. Furthermore, we let FRESCO use the static estimator to provide estimates for the same time intervals. Figure 3 shows the MAPE on the y-axis and the length of the time interval on the x-axis, for each of our three datasets. A value of 30% at the interval length of 1 minute for the laptop dataset means that, on average, the energy consumption estimates summed up for intervals of 1 minute deviate by 30% from the corresponding consumption measured. Figure 4 shows the MaxAPE for all datasets. The figures indicate that for longer time intervals, FRESCO can provide more accurate estimates. In particular, for the server dataset, the maximum error goes from around 23% for an aggregation level of one second to around 11% for a higher aggregation level of 16 minutes, corresponding to a two-fold decrease in value. Smaller decreases (69–62% and 31–30%) occur for the other two datasets. This is because longer interval lengths mitigate the effect of short-term deviations in the workload. Estimates depend on the operator approximation of the average load. Thus, the results for the laptop dataset show that the operator has under- or overestimated the average load significantly. Evidently, the accuracy of the estimation can be improved if the operator provides more accurate information on the workload of the target system.

Summary: *FRESCO has provided upper bounds for the energy consumption. Furthermore, it is able to provide reasonable estimates for the average workload by requiring only little data from the operator. Thus, we conclude that FRESCO is able to deal with the requirements of this use case.*

2) Demand Response: This use case (cf. Subsection II-2) requires time series of estimates to identify periods of time with high energy consumptions (peaks), together with upper and lower bounds. As the operator is willing to invest only a small effort, FRESCO suggests our dynamic estimator model.

To evaluate this scenario, we let FRESCO estimate (cf. Equation 6) the consumption based on the CPU load and on information on the maximal and minimal energy consumptions of our three target systems with a frequency of one second. We use these estimates to identify points in time when the energy consumption is above a given threshold. In particular, we evaluate two dynamic thresholds that consider the difference between the largest and smallest values of a time series T:

$$\theta_1 = 0.8 \cdot \big(\max_{i=1}^{|T|}(T_i) - \min_{i=1}^{|T|}(T_i)\big) \tag{8}$$

$$\theta_2 = 0.95 \cdot \big(\max_{i=1}^{|T|}(T_i) - \min_{i=1}^{|T|}(T_i)\big) \tag{9}$$

We compute time series of peak consumption from our measured values as well as for the time series FRESCO has estimated, by filtering out all values that are smaller than θ. If our estimates are accurate, FRESCO can identify periods with high energy consumption and can thus enable operators to perform Demand Response.

Figure 5 illustrates the cumulative distribution function (CDF) of the real energy consumption during specific intervals for the desktop dataset. The first set of intervals is when FRESCO estimated the consumption to be greater than θ_1 (continuous line). The second set is when FRESCO estimated the consumption to be greater than θ_2 (dashed line). We observe that, if the estimator predicts a value greater than

θ_1, then the real energy consumption is greater or close to θ_1. Thus, in around 88% of all cases, a value predicted to be greater than θ_1, is also greater than θ_1. In 80% of all cases where a value greater than θ_2 was estimated, the real energy consumption was greater than 90% of θ_2. On the other hand, our estimator predicted a value of at least 75% of θ_1 for intervals with consumptions greater than θ_1 in 90% of the cases. θ_2 has produced similar results.

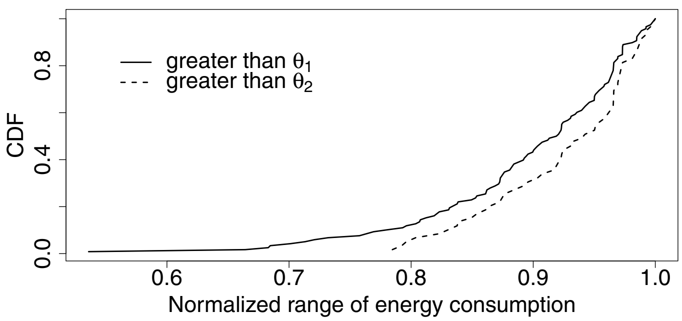

Fig. 5: CDF, Desktop Dataset

We have obtained similar results for our laptop dataset. Concerning the server dataset, for which we use minute-by-minute measurements and estimations, the accuracy of our dynamic estimator is better. In 90% of the cases where an estimate is greater than θ_1, the real value also is greater than θ_1 of the real energy consumption. Additionally, the estimator correctly predicted all values greater than θ_2.

Summary: *Our dynamic estimator can identify periods of time with peak energy consumptions with a reasonable accuracy with a low estimation effort. That is, it uses only static information on the minimal and maximal consumption of the target system, and it samples only the CPU load. Moreover, the estimator is flexible, i.e., it can sample more usage parameters in order to improve its accuracy. We conclude that FRESCO fulfills the requirements of this use case.*

3) Energy Accounting: Our third use case (cf. Subsection II-3) requires estimates with stochastic guarantees. Thus, FRESCO suggests a calibration-based estimator, which provides estimates based on calibrated energy profiles.

We let FRESCO calibrate energy profiles for each of our three target systems at setup time. At run time, our monitoring application samples the CPU load with different sampling frequencies. In order to compare the estimates with the real values, we have calculated MAPE and MaxAPE for all three datasets. Figure 6 shows the MAPE on the y-axis and the length of the time interval on the x-axis. The figure indicates that the estimation accuracy is better for longer time intervals. For all three datasets, the mean error decreases by around a fourth (14–37% decrease) when the estimator aggregates estimates for intervals of 16 minutes instead of one second. Similarly, the MaxAPE decreases significantly for all datasets with longer estimation intervals, as shown in Figure 7. In particular, for the server dataset, the maximum error is around 6.5 times smaller, decreasing from about 35% to 5.3%.

Summary: *With estimation intervals that make sense in energy accounting, FRESCO is able to provide estimates of a high accuracy. While the effort at run time is similar to the one of the dynamic estimator, the effort at setup time is very high. However, many energy accounting scenarios make use of numerous target systems with identical hardware, e.g., for typical office tasks. In such scenarios, the calibration effort at setup time takes place only once. Thus, we conclude that FRESCO can perform well in an energy accounting scenario.*

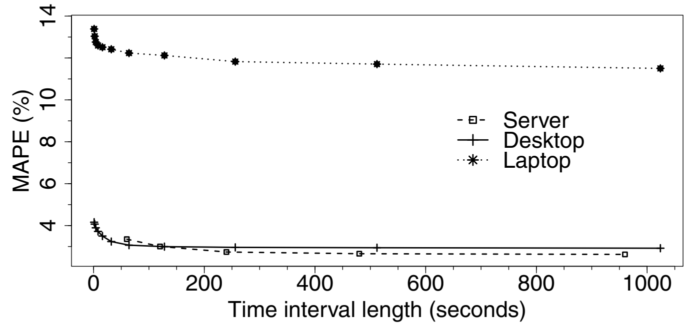

Fig. 6: MAPE, Calibration-Based Estimator

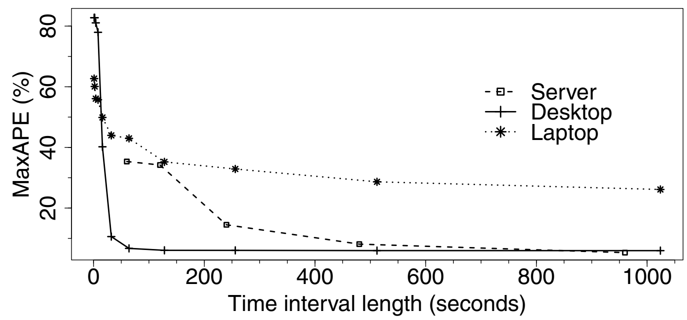

Fig. 7: MaxAPE, Calibration-Based Estimator

VI. Related Work

The energy consumption of a computing system can be monitored directly. This is done by measuring the energy consumption using common digital meters [2], custom-designed devices [3] or integrated hardware power sensors [20]. In the case of large and heterogeneous computing centers, installing digital meters at every subsystem (server, PC, etc.) is costly.

A related domain of interest in recent research is computer power characterization at (sub)system level. Part of recently developed power characterization models are based on collecting microarchitectural events using hardware registers. These models consider both subsystem [21], [22], [23] and system [24] levels, as well as virtualized environments [25]. However, taking hardware registers into account makes power models less portable. In the case of large heterogeneous deployments of computing systems, this lack of generality increases the effort for power consumption modeling and estimation of the entire deployment. Another issue is that the number of hardware performance events tends to be large [26]. Moreover, only a small part of them can be measured at the same time [26], due to the limited number of hardware registers. A solution is to time-multiplex different sets of events on the hardware registers [27]. While this approach allows for a greater number of performance events to be monitored, it increases the overhead and reduces the accuracy.

Using high-level statistical information provided by the operating system avoids the need for specific detailed (low-level) hardware knowledge when designing power estimation models. Recent work has proposed power consumption models based solely on high-level performance or usage metrics provided by the operating system to maximize energy efficiency using various optimizations. Thus, Fan et al. [11] uses CPU utilization in order to estimate the power consumption of large numbers of servers, reaching a mean error of 1% when considering groups of several hundreds of servers. The power consumption model proposed in [28] uses the expected load on a server cluster in order to estimate its power consumption. $JouleMeter$ [29] is a solution for virtual machine power metering which infers the power consumption from resource usage at runtime. A model of the power consumption of idle servers is proposed in [30].

The advantages of high-level black-box models are the low overhead, simplicity and relatively good accuracy. However, these models estimate full-system power consumption and do not allow for a more fine-grained repartition of the power consumption, such as per process or per application power consumption. Moreover, the majority of the models has been developed and tested on computer systems with a big share of static energy, such as servers [11] or clusters of virtual machines [29]. In order to model such computer systems, these black-box models are easily integrable into FRESCO.

VII. Conclusions

As the share of computer energy consumption increases, it becomes increasingly important to quantify it in a solid manner. Many enterprise applications, accounting procedures and business models require such data to allow informed management decisions, e.g., for energy-aware management of IT resources, IT-energy accounting or demand response. However, most existing estimators are tailored to specific use cases, hardware architectures and usage profiles. Due to their different characteristics in terms of effort and accuracy, the choice of estimation method for a given application is far from obvious. In this article, we have proposed FRESCO, a general and flexible FRamework for the Energy eStimation of COmputers. Depending on the effort the operator is willing to invest and on the requirements of the application, FRESCO can propose and run appropriate estimators with good parameter settings. It gives quality guarantees on the estimates. FRESCO considers heterogeneous hardware components and loads, as well as the frequency of the estimation. Experimental results based on three representative real-world datasets show that our framework is useful in many business cases.

References

[1] W. Vereecken et al., "Overall ICT Footprint and Green Communication Technologies," in *International Symposium on Communications, Control and Signal Processing (ISCCSP)*, 2010.

[2] R. Ge et al., "PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 21, 2010.

[3] D. C. Snowdon, S. M. Petters, and G. Heiser, "Power Measurement as the Basis for Power Management," in *Workshop on Operating Systems Platforms for Embedded Real-Time Applications*, 2005.

[4] M. Poess and R. O. Nambiar, "Power Based Performance and Capacity Estimation Models for Enterprise Information Systems," *IEEE Data Engineering Bulletin*, vol. 34, 2011.

[5] A. Noureddine et al., "Runtime Monitoring of Software Energy Hotspots," in *IEEE/ACM International Conference on Automated Software Engineering*, 2012.

[6] S. Rivoire, P. Ranganathan, and C. Kozyrakis, "A Comparison of High-Level Full-System Power Models," in *HotPower*, 2008.

[7] A. Kansal, F. Zhao, J. Liu, N. Kothari, and A. A. Bhattacharya, "Virtual Machine Power Metering and Provisioning," in *ACM Symposium on Cloud Computing*, 2010.

[8] S. Rivoire et al., "JouleSort: A Balanced Energy-Efficiency Benchmark," in *ACM SIGMOD International Conference on Management of Data*, 2007.

[9] J. Laudon, "Performance/Watt: The New Server Focus," *ACM SIGARCH Computer Architecture News*, vol. 33, 2005.

[10] M. Milenkovic, E. Castro-Leon, and J. R. Blakley, "Power-Aware Management in Cloud Data Centers," in *International Conference on Cloud Computing (CloudCom)*, 2009.

[11] X. Fan, W.-D. Weber, and L. A. Barroso, "Power Provisioning for a Warehouse-Sized Computer," in *Annual International Symposium on Computer Architecture*, 2007.

[12] P. Palensky and D. Dietrich, "Demand Side Management: Demand Response, Intelligent Energy Systems, and Smart Loads," *IEEE Transactions on Industrial Informatics*, vol. 7, 2011.

[13] A. Berl et al., "Modelling Power Adaption Flexibility of Data Centres for Demand-Response Management," in *Energy Efficiency in Large Scale Distributed Systems*, 2013.

[14] V. Jimenez et al., "Energy-Aware Accounting and Billing in Large-Scale Computing Facilities," *IEEE Micro*, vol. 31, 2011.

[15] P. Efros, E. Buchmann, and K. Böhm, "FRESCO: A Framework for the Energy Estimation of Computers (Extended Version)," Karlsruhe Institute of Technology (KIT), Tech. Rep., 2014. [Online]. Available: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000039158

[16] Accessed 30 April 2014. [Online]. Available: http://www.tpc.org/information/benchmarks.asp

[17] R. Basmadjian and H. de Meer, "Evaluating and Modeling Power Consumption of Multi-core Processors," in *International Conference on Future Energy Systems (e-Energy)*, 2012.

[18] D. Economou, S. Rivoire, and C. Kozyrakis, "Full-System Power Analysis and Modeling for Server Environments," in *Workshop on Modeling Benchmarking and Simulation (MOBS)*, 2006.

[19] Accessed 30 April 2014. [Online]. Available: https://www.wattsupmeters.com/secure/products.php?pn=0&wai=638&spec=8

[20] Intel, "Intelligent platform management interface," accessed 30 April 2014. [Online]. Available: http://www.intel.com/design/servers/ipmi/index.htm

[21] J. Janzen, "Calculating Memory System Power for DDR SDRAM," *Designline*, vol. 10, 2001.

[22] A. Merkel and F. Bellosa, "Balancing Power Consumption in Multiprocessor Systems," in *ACM SIGOPS/EuroSys European Conference on Computer Systems*, 2006.

[23] R. Bertran et al., "A Systematic Methodology to Generate Decomposable and Responsive Power Models for CMPs," *IEEE Transactions on Computers*, vol. PP, 2012.

[24] W. L. Bircher and L. K. John, "Complete System Power Estimation Using Processor Performance Events," *IEEE Transactions on Computers*, vol. 61, 2012.

[25] G. Dhiman, K. Mihic, and T. Rosing, "A System for Online Power Prediction in Virtualized Environments Using Gaussian Mixture Models," in *Design Automation Conference*, 2010.

[26] S. M. Rivoire, "Models and Metrics for Energy-efficient Computer Systems," Ph.D. dissertation, Stanford University, 2008.

[27] R. Azimi, M. Stumm, and R. W. Wisniewski, "Online Performance Analysis by Statistical Sampling of Microprocessor Performance Counters," in *Annual International Conference on Supercomputing*, 2005.

[28] T. Heath et al., "Energy Conservation in Heterogeneous Server Clusters," in *ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, 2005.

[29] A. Kansal et al., "Virtual Machine Power Metering and Provisioning," in *ACM Symposium on Cloud Computing*, 2010.

[30] R. Basmadjian, F. Niedermeier, and H. De Meer, "Modelling and Analysing the Power Consumption of Idle Servers," in *Sustainable Internet and ICT for Sustainability (SustainIT)*, 2012.