Article:
Kavanagh, R orcid.org/0000-0002-9357-2459 and Djemame, K orcid.org/0000-0001-5811-5263 (2019) Rapid and accurate energy models through calibration with IPMI and RAPL. Concurrency Computation Practice and Experience, 31 (13). e5124. ISSN 1532-0626

https://doi.org/10.1002/cpe.5124

Received 26 April 2016; | Revised 6 June 2016; | Accepted 6 June 2016

DOI: xxx/xxxx

ARTICLE TYPE

Rapid and Accurate Energy Models through Calibration with IPMI and RAPL

Richard Kavanagh* | **Karim Djemame**

[1]School of Computing, University of Leeds, Leeds LS2 9JT West Yorkshire, UK

Correspondence
*Corresponding author name, This is sample corresponding address. Email: r.e.kavanagh@leeds.ac.uk

Summary

Energy consumption in Cloud and High Performance Computing platforms is a significant issue and affects aspects such as the cost of energy and the cooling of the data centre. Host level monitoring and prediction provides the groundwork for improving energy efficiency through the placement of workloads. Monitoring must be fast and efficient without unnecessary overhead, to enable scalability. This precludes the use of Watt meters attached per host, requiring alternative approaches such as integrated measurements and models. IPMI and RAPL are subject to error and partial measurement, which may be mitigated. Models allow for prediction and more responsive measures of power consumption, but require calibrating. The causes of calibration error are discussed, along with mitigation strategies, without overly complicating the underlying model. An outcome is a Watt meter emulator that provides hosts level power measurement along with estimated power consumption for a given workload, with an average error of 0.20W.

KEYWORDS:
power, energy, IPMI, RAPL, energy model, calibration

1 | INTRODUCTION

Energy efficiency in both High performance computing (HPC) and Cloud data centres is fast becoming a primary concern. It has a significant impact upon the running costs, environmental impact and cooling of data centres and is a barrier towards the exascale. Data centres are therefore placing an ever increasing importance on attempts to save on energy consumption at various levels of abstraction. Accurate and timely information regarding power consumption is hence important in establishing ways to mitigate both the energy consumed and the overall cost. This enhanced information especially at host level can be used by middleware such as Cloud infrastructure managers e.g. OpenNebula[1], OpenStack[2] and Resource and Job Management Systems (RJMS) e.g. SLURM[3].

To this end accurate monitoring and prediction can be considered as the fundamental groundwork for improving the energy efficiency of such environments. This monitoring is required to be fast and efficient without unnecessary overhead, as well as being able to scale to the size of a data centre. This scale means measurement through directly attached Watt meters is unrealistic. This therefore requires alternative approaches such as integrated measurement or models that translate resource utilisation into the power consumed by a physical host. Integrated measurement approaches include methods such as Intelligent Platform Management Interface (IPMI)[4] and Running Average Power Limit (RAPL)[5].

IPMI[4] is a message-based, hardware-level interface specification which operates independently of the operating system. It works through interacting with a baseboard management controller (BMC), which is a micro-controller embedded on a computer's motherboard that is used to collect data from various sensors. It is used by system administrators principally for recovery procedures or monitoring platform status (such as temperatures, voltages, fans, power consumption, etc.). IPMI can be found in nearly all current Intel architectures and if power sensors are supported it can provide a very cheap inbuilt way to collect data[6]. Various open-source software solutions exist for inband and out-of-band collection of IPMI sensors data[7,8]. Hackenberg et al.[9] show that the power data collected from IPMI is reasonably accurate but the calculation of energy consumption

might not be precise for jobs with small durations as well as jobs with regular power variations. Models in these situations have advantages given the inputs to the model and hence outputs can be more responsive than IPMI, thus can accommodate fluctuations.

RAPL[5] provides operating system access to energy consumption information based on a software model driven by hardware counters. This model tracks the energy consumption of the CPUs, integrated GPU and DRAM[10]. This can be done at the level of package and per core, with processors aimed at desktops focus on package core and GPU, while server market processors focusing on package, core and DRAM domains[9,11]. It is widely available in Intel Sandy Bridge and later processors[5]. RAPL is reported to measure energy consumption reliably in[10,12], though it should be noted it does not measure the energy consumption of all of the physical host. It is therefore only a reliable measure of a portion of a host's overall energy consumption. PAPI[13] performance profiling tool is one such tool that supports monitoring energy consumption through RAPL interfaces.

Integrated measurement through IPMI, RAPL or directly attached Watt meters can only be used for current consumption and cannot be used for prediction and hence forward planning. Models allow for prediction which aids adaptation and forward planning, but require calibrating and are hence subject to the original sensor error, that should be minimized where possible. Otherwise these sensor inaccuracies would diminish the overall accuracy and usefulness of the energy models. Simple models with reduced error from sensors is the cornerstone of this paper, through means of reducing noise at calibration time. Once calibrated these models can then be used in order to predict future power consumption, give more rapid estimates of power consumption or may be used to attribute power consumption to energy users such as Virtual Machines (VMs) or applications (including short lived executions).

Focus is therefore placed on the sources of this error and consider how it can be mitigated. This mitigation of error is illustrated in tools that have been created which enable the measurement of the energy efficiency of service deployments in both Cloud and HPC based environments. The overall aim of this energy modelling tooling is to model, measure and report on energy efficiency for both billing and reporting purposes. The reporting can then be used to assist developers in understanding and minimising their overall energy consumption, including in practical situations where sensor accuracy may be limited.

This paper's main contributions are:

- identification of the challenges for accurate and fine-grained power and energy measurements and evaluate opportunities and limitations regarding integrated node power measurements.
- a comparison between IPMI and RAPL gathered power measurements vs Watt meter measurements with discussion regarding accuracy for energy models.
- recommendations on how to calibrate energy models, with the aim of reducing error.
- discussion regarding actual error in energy models and the causes of such error.
- an illustration of the use of segmented linear regression as a means to overcome non-linearity in power vs CPU utilisation which avoids over fitting calibration data with high order polynomials.
- discussion regarding the transposition of the proposed techniques towards accelerators such as GPUs.

The remaining structure of the paper is as follows: The next section covers the related work, discussing the background of internal measurement sensors and energy modelling. Energy measurement is considered in both Cloud and HPC environments, thus the challenges of data collection are discussed in Section 3. Recommendations are given on how to meet these challenges in Section 4. The process of modelling is discussed in Section 5 which is split into two principle elements. The first covers the process by which good quality calibration data can be obtained for the models (Section 5.1). The second part discusses post-calibration and mechanisms for allocating power consumption to virtual machines in Section 5.3. An evaluation is then performed in Section 6 discussing the accuracies of both IPMI and RAPL based power sensors in a HPC based environment (Section 6.3.1) and how they might be utilised collaboratively to calibrate a model. This work is further extended in a Cloud environment where IPMI is compared to direct measurement with a Watt meter again considering a model based approach in Section 6.3.2. The lessons learned are summarised in Section 7 and the paper is concluded with discussion of future work in Section 8.

2 | RELATED WORK

The characterisation of the resources is an important step in regards to accurate energy predictions for software usage. This gives rise to profiling and testing frameworks such as JouleUnit[14] that enable the profiling of hardware systems in order to understand their power consumption profiles. Profiling physical resource for power consumption in distributed systems, require more generic distributed monitoring frameworks such as Zabbix[15] or others[16].

Data for resource's power consumption is principally obtained either by direct measurement [17] or inferred via software and physical performance counters [14,18]. Direct measurement obtains the wall power [18] value via the use of Watt meters [17], providing an aggregation of the current power usage of a physical resource [19]. Direct measurement can be difficult as it needs specialised equipment such as Watt meters including the prospect of meter integration such as PowerInsight [20] and PowerPack [21] for power consumption of node's individual components. These measuring processes are accurate but in utilising additional hardware can be costly and cause difficulties in scaling to the size of a data centre.

Performance counters [22,12,23] are a non-invasive means of determining energy usage, by utilising counters which are located within the CPU and Operating System. Wall power measurements have the advantage of accuracy but require the specialized physical hardware to be attached onto the infrastructure, while the performance counters are indirect measures of power consumption and require a model to derive an estimate of the energy consumed. IPMI offers the potential for direct measurement of power consumption based on sensors integrated into the physical host, thus it is non-invasive like performance counters and offers the potential for high accuracy as well, although this accuracy is not always realised, thus models are required.

In order to determine VM or host energy usage various frameworks have been developed. The majority of cases use linear models [18,22,24,25], which is shown here to not always be representative of what actually occurs in real systems. Schubert et al. [26] remark how easy it is to get calibration wrong with such models especially when averaging or aggregation is used. In most cases linear models have provided power estimates with a high degree of accuracy for VMs and their underlying resources, usually within 3W of the actual value or within 5% error. Additive models such as [18,24] utilise load characteristics for each of the major physical components such as CPU, disk and network, each of which is considered separately and summed together. In these cases idle power consumption is treated as an additional model parameter that is simply added to the other load characteristics. Not all models are purely additive, for example two linear regression models which are then merged using a bias mechanism may be used [22]. The first model covers power for CPU and cache and the other for DRAM and disk. An alternative approach is to use principle component analysis to learn the importance of each parameter in the model, which again avoids additive models [23]. The use of performance counters can also differ amongst existing models, such as physical meters being only needed during an initial training phase followed by the use of counters post training [27].

The second concern after profiling a physical host's power consumption is to determine its future energy consumption, which can then be used to guide both the deployment and operation of the environment as a whole. Estimating future energy consumption requires an understanding of the system's workload over time. This can include CPU load prediction in models such as LiRCUP [25] which is aimed at assisting in the maintenance of Service Level Agreements (SLAs) and others [28,29] that search for workload patterns. Workload prediction has enjoyed a lot of attention with a particular focus on the cloud property of the scaling of resources and the maintenance of Quality of Service (QoS) parameters [30,31,32,33]. Workload prediction in Clouds has also been seen as a means to plan future workloads so that physical hosts may be switched off when not required [34], but may also be used as a basis of the prediction of future power consumption.

Trends in power models are towards more increased complexity due to heterogeneous systems (FPGAs, GPUs etc) and aspects such as the utilisation of DVFS. The focus in this paper is the elimination of initial noise in the calibration dataset yet these aspects remain important. In Adhinarayanan et. al. [35] online power estimation of GPUs is considered, its narrow focus on GPUs and not a full host limits however limits its use in a wider context. To its particular advantage it considers instantaneous power instead of average power over a longer period of time, which makes it particularly useful for use cases such as adaptation and application monitoring. LPGPU^2 [36] consider GPU application developers providing power measurement devices to better understand power consumption and bottlenecks within the context of GPU accelerators, which is a useful outcome of modelling, especially in cases with short execution times [12]. Song et. al. [37] utilises Neural Networks as a means establishing estimates on GPUs, while GPUWattch [38] consider a simulation based approach which has the drawback of requiring a detailed knowledge of the underlying GPU architecture. Sundriyal et. al. [39] considers DVFS for both memory and CPUs, as a means of improving the efficiency of computation, the power modelling remains limited to utilising RAPL. Any model that considers p-states and the model calibration process considered here could be combined, enabling greater accuracy. DVFS is also considered in [40] in the context of CPU-GPU hybrid clusters, like [39] focusing on scheduling within these heterogeneous environments with power as a key aspect.

This work focuses on errors introduced by measurements during calibration, but for long term predictions, accurate estimates of workload are very important. Kwapi [41] is the most closely related monitoring tool to our own work, in that it focuses on power and energy monitoring, however our framework expands on this to both VMs (Clouds) and applications (HPC). The most closely related work in terms of measurement which considers sources of error is [9,42], though the focus in ours is upon the training of models as opposed to a focus purely upon measurement accuracy.

3 | THE CHALLENGES WITH POWER AND ENERGY DATA COLLECTION

In order to gather sensor data in generalised monitoring infrastructures e.g. Zabbix [15], Ganglia [43] etc. there are various issues that need to be overcome. The sources of data and quality of data returned need to be understood. Data for example may come from many sources such as the

operating system, which can utilise special structures such as /proc/ on Linux, or via more specialist hardware such as baseboard management controllers (BMCs) and standardised interfaces. This can include aspects such as CPU performance counters as well as standardised interfaces, such as IPMI.

Dependant on the purpose of the data and the properties that can be attributed to it various considerations need to be made. Data gathered by such sensors can be utilised to either measure power/energy directly or generate a model, that can calculate the power consumed based on resource utilisation. Errors can be introduced to models in two phases. The first is the calibration phase and the second is at operation time. The models can then be used for example to drive an energy modeller for determining application and VM power consumption or Watt meter emulator.

The calibration phase results in an inaccurate model that does not correctly represent the relationship between load and power consumption. This can occur for several different reasons, which are associated with how the measurement is taken. These aspects of measurement affect both calibration and general measurements at operation time. These issues are namely:

Unsynchronized metric update intervals for different metric types: This could occur when measuring CPU utilisation and power together. For calibration to be accurate it requires the measurements to be perfectly synchronized or for the utilisation to remain stable during a measurement phase, so that both measurements represent the physical host in the same state. This avoids one measurement arriving yet the other one being relatively stale. At operation time models that infer power consumption automatically update their reported values once new measurements arrive, thus mitigating some synchronization issues.

Measurement arrival latency (Monitoring infrastructure overhead): Differing from the above case, where synchronisation issues may occur, this is caused by the inherent delays in taking a measurement, transferring the value across a network and recording it in the monitoring infrastructure. This affects the detection of the start and end of periods of induced load, particularly when monitoring the most recent metric values to arrive, as opposed to a historic trace of measurement values. This can be seen in cases where measurements have no associated time stamp in that changes in CPU utilisation are reflected in the power consumption slightly afterwards, rather than instantaneously. This issue can be mitigated either by synchronising clocks and taking time stamps for each data item or by performing the calibration run locally without the use of a full monitoring infrastructure, such as Zabbix, Ganglia etc. Such local monitoring only works during the calibration and will not work during operation time and additionally has the side effect of measuring a small amount of additional load induced by the monitoring.

Averaging and time windows of measurements' values: Measurements arrive with a given polling interval. However measurements such as CPU load also have a time window in which the measurement was taken e.g. over the last minute. This averaging causes errors in the model and requires the CPU utilisation measurement window to be made as small as possible. One alternative is for measurements used in the calibration dataset to only start to be taken after load has been induced for a time that is longer than the length of the averaging period. The former option is simpler but requires custom scripts in the case of the Zabbix monitoring environment. At operation time the shorter time window is especially important when monitoring individual applications, as it would otherwise blur the start and end times for monitoring the application. In the case of CPU utilisation as an input into the power model the follow-on implications for the power metrics resolution would also have to be considered.

Update interval of a sensor's reported value: Sensors such as power measurements taken over IPMI update slower than the interval at which the baseboard can be queried. Thus rapid polling of the interface can result in the previously reported value being reported again, without prospect of change. Hence the poll interval should not exceed this update interval, e.g. In the case of IPMI power values polling might be restricted to every 5 seconds.

Sensor Resolution: Sensors are subject to measurement error and have confidence values associated with them. An example of this is Watt meters such as a WattsUp Meter Pro[44] which have an accuracy of +/- 1.5%. IPMI uses only 1 byte of data to represent a sensor reading[4,45]. This results in lower resolution on power consumption which is reported in[46] of having increments of every 14W, although this is dependant upon being able to represent the range of power values possible with 1 byte. To circumvent this resolution issue, some vendors (e.g., Dell and Hewlett Packard[47,45] implement proprietary extensions to the IPMI protocol. Resolution also can also be within the temporal domain associated with a measurement, RAPL for example lacks timing information associated with every updated of the energy counter[9], making it more difficult to accurately profile applications.

Partial Observability and Metric Quality and instrumentation issues: Not all metrics that can be gathered to express system utilisation are necessarily proportional to power consumption and only provide a partial view of the system. Some metrics require extra instrumentation such as running code in a debugging mode that complicates and negates generalisability. This challenge can be expressed as an example in regards to GPU utilisation. Nvidia GPUs for example report utilisation as the percentage of time in a given interval where it is true that at least one stream multiprocessor (SM) is active[48], which has a limited direct correlation to power consumption, given a workload can use 1 or N stream multiprocessors and still report the same utilisation level. It equally is less representative during transitions between loaded and unloaded states in regards to power consumption.

Alternatives may be considered such as using the clock frequency of a stream multiprocessor instead. If the GPU is inactive, then, the value will be low, otherwise if it is active the cores frequency will scale to a high value.

Calibration Workload Pattern Generation: The generation of workloads that are suitable for determining accurate trends between workload and power utilisation (see section 5.1) can be difficult. This can be shown in cases such as graphics cards, where the device has several p states, that must be explored. Workload on GPUs is also arranged into a series of warps by the GPUs own scheduler, so attempting to select a specific fixed level of workload can be more difficult.

4 | RECOMMENDATIONS

The issues with data collection shown in Section 3 provide the basis of several recommendations which are implemented in this paper. They are to be considered in calibration phases such as in Section 5.1 or using the sensors directly for measurement. The recommendations are as follows:

- to use metrics that represent the physical host in its most recent state, referred to here as spot metrics and tend to avoid averaging and representing long periods of time.
- that load should be induced followed by waiting a set period of time for the values to stabilise and then take measurements. This period of time is to be at least greater than any averaging window used on the power and utilisation metrics. A further addition to this is to detect plateaus in the measured values and only using congruent data points, which can be used as a mechanism to determine how long to wait before accepting measurements as being valid.
- to obtain measurements locally thus avoiding monitoring system overheads including network delays.

At calibration time, delays in the arrival time of measurement data or averaging recent utilisation data can have dramatic effects on the calibration's accuracy. At operation time averaging in some cases can be useful, which is highly dependant on the purpose of the output data. Averaging over a time window for a measurement can be utilised to generate a smoothing effect on the data at the cost of responsiveness and overall accuracy. Averaging softens changes in values where rapidly fluctuating estimated power consumption values might influence the decision about a deployment's positioning. In other cases such as measuring an individual applications power consumption with the intent of re-factoring code, requires more responsive output data, with a smaller measurement window of utilisation, that drives the energy model.

5 | ENERGY MODELLING

Energy modelling has several key functions within a Cloud or HPC environment. The first is the discovery of the amount of energy consumed where it cannot be directly measured, this includes: forecasting of future power consumption and for individual applications and virtual machines. The second regards the adaptation to power and energy values with the aim of mitigating the energy consumed. These models are realised within a monitoring framework as the Energy modeller and the Watt Meter emulator[49,50].

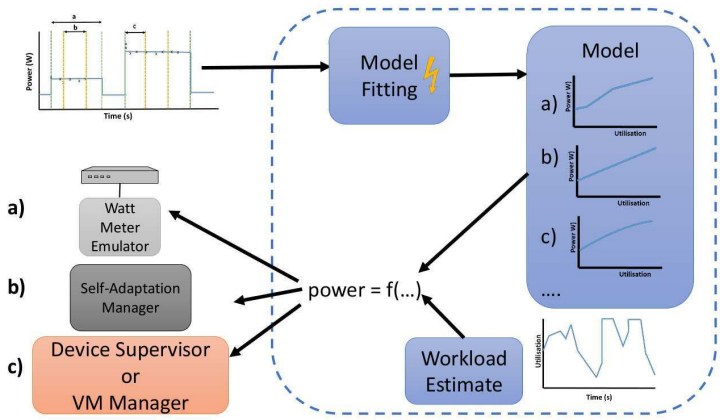


FIGURE 1 The overall flow for calibration and model usage

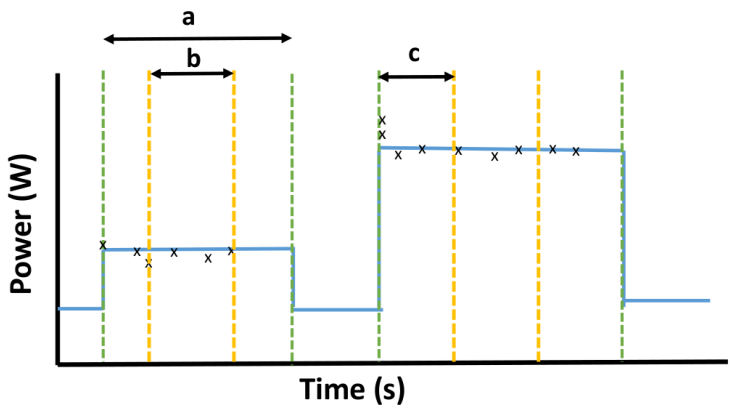


FIGURE 2 The construction of artificial traces for calibration

Energy modelling has several key phases (Figure 1), the first is calibration which is discussed next consisting of data gathering and fitting and the second is at runtime when models can allocate power consumption to VMs or applications. The latter data being used generally to drive systems such as scheduling (VM Manager/Device Supervisor), Information reporting (Watt Meter Emulation) or self-adaptation.

5.1 | Model Calibration

The calibration data gathering process aims to create standard repeatable conditions that generate a sequence of precise loads on the physical host undergoing measurement. The aim is to tightly control the environment while running an experiment to gain an accurate mapping between the resource utilisation and power consumption. If consistent load can be maintained this results in high quality calibration data that maps utilisation to power consumption accurately. This overcomes many of the problems of gathering power and energy metric data as discussed in Section 3. This data can then be used as the basis of predicting future power consumption/energy usage, as well as providing faster and more responsive measures of current power consumption especially for short runs of an application.

The calibration data gathering process works by inducing load on the physical host, while limiting the load to a fixed utilisation level. The fixed load then runs for a period of time (see Figure 2). A sequence of plateaus are shown with ever increasing utilisation. In between each plateau is a small gap where no load is induced and the CPU returns to its idle state. The duration of each run (marked as (a)) can be extended as required. This is particularly useful in cases where measurements are averaged and the values reported represent a time window. A longer time period (a) gives a greater chance that the reported utilisation and power level stabilises, which means that the average only represents a period of time during this plateau. In addition a longer time period (a) will provide more data points per load period. Not all datapoints can be treated as valid as each other however. Issues such as the metrics being recorded are not synchronised or if the arrival of the values undergo delay due to network, or other mechanisms such as caching can all have their effects. A key solution to this is to discard values at the start and end of an experimental run (indicated by (c) in Figure 2), as a means to overcome these issues. In addition to this, as soon as the load is induced the measured load may show a spike above the intended target load, thus inducing experimental error. The final set of datapoints in the area indicated by (b), thus can be used for calibration data, given these points are not subject to error. If metrics such as power or CPU utilisation are subject to averaging discarding values so that the time (c) is at least the length of the averaging window is useful. It means the datapoints in the area (b) only represent values at the fixed workload and is a means of discounting measurement inconsistencies.

After the calibration data has been collected the energy modeller can then perform curve fitting. Curve fitting can then be applied with any of the following fits:

1. linear
2. polynomial
3. spline polynomial

In the case of polynomial only low order polynomials are used to avoid over fitting the data. The applicability of each curve is tested with the goodness of fit measure Root Mean Square Error (RMSE). This allows for the automatic selection of the most suitable fit for a given host, thus ensuring enough flexibility in a heterogeneous infrastructure. In this paper both linear and spline polynomials are used.

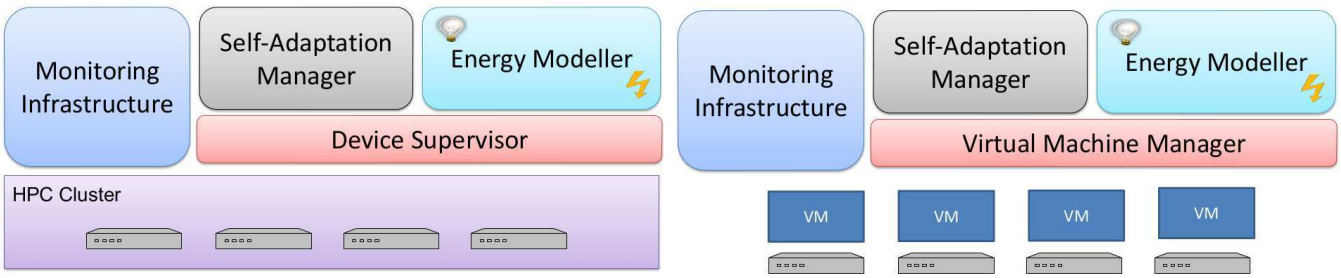
In regards to calibration it can additionally consider accelerators such as GPUs, though it should be understood the main effort of the paper considers IPMI and RAPL and how the quality of metrics obtained affects model accuracy. The ideal situation for calibration with accelerators remains the same as the CPU, i.e. by inducing fixed loads and determining the power consumed. Following a strategy of simplicity two sub-models can be used to handle accelerators. Principally curve fitting is considered for the CPU followed by a separate sub-model for GPUs or other accelerators.

Two generic strategies are implemented within the energy modeller. The first is a bi-modal predictor that determines power usage of an accelerator assuming an unutilised and heavily utilised state and clustering data into one of these two states. Thus the load for prediction on a GPU only needs to indicate the systems state, in order to obtain a prediction. This is useful in cases where the quality of calibration data is poor, as it still offers an estimate that can give a guide processes such as scheduling or self-adaptation. It may be recalled in Section 3 a key issue was the generation of calibration workloads in accelerators.

The second generic additive model of the energy modeller, considers the CPU and the accelerator's utilisation separately. The CPU can be treated as before while a multilayer perception network with a single hidden layer, can be used for the GPU (or any other arbitrary accelerator with sufficient high quality utilisation and power metrics). The amount of inputs is based upon the size of the calibration data gathered providing a single output. The size of the hidden layer is scaled to be $\sqrt{\text{inputsize} + \text{outputsize}}$, aiming for a size that is sufficient but not so large as to cause it to be overly trained. The emphasis is therefore placed upon gathering training data of sufficient quality for the network to train correctly, ensuring that the parameters chosen have sufficiently strong influence on the power consumption. Key also to this is that it should remain practical, so attaching profiling to every application running in order to obtain performance counters for each application is not suitable.

Finally once the model is calibrated, even if the power measurement sensor reports an average value, an instantaneous estimated power consumption value can be obtained without averaging and at a higher temporal granularity by using the model instead of direct measurement.

5.2 | Runtime Usage of Modelling



(a) HPC **(b)** Cloud

FIGURE 3 Usage of energy monitoring information in Cloud and HPC use cases

One aspect of energy modelling is its usage once the model has been constructed. Figure 3 illustrates the two main contexts to which an energy model may be used. Figure 3a shows the energy modeller in a HPC based environment and Figure 3b in a cloud based environment. In HPC case the device supervisor manages the infrastructure, which is a range of physical bare metal devices. It is supported in cases where deployments are energy aware by the energy modeller. The device supervisor in this case when performing scheduling queries the energy modeller for the projected power usage of the deployment. At runtime a self-adaptation manager which can be used to manage QoS goals of the applications can utilise the energy modeller to gain current power consumption information for applications that are running[51]. This is achieved through the interaction of the energy modeller and the monitoring infrastructure. The monitoring infrastructure provides information about the physical hosts such as utilisation and power consumption, while the energy modeller determines the power consumption of each application. This partnership between the monitoring infrastructure and energy modeller continues post runtime, in terms of billing power usage to a given application. In Figure 3b a Cloud based environment is illustrated, this bares similarities to the HPC case. The major change is in that physical hosts are replaced with virtual machines. These virtual machines need their power monitoring and the energy modeller focuses upon them instead of the applications. The device supervisor is replaced by the virtual machine manager, which like the device supervisor when using energy aware deployments can utilise the energy modeller to understand the likely impact of any given deployment. The remaining components in the Cloud architecture work in a similar fashion to the HPC use case.

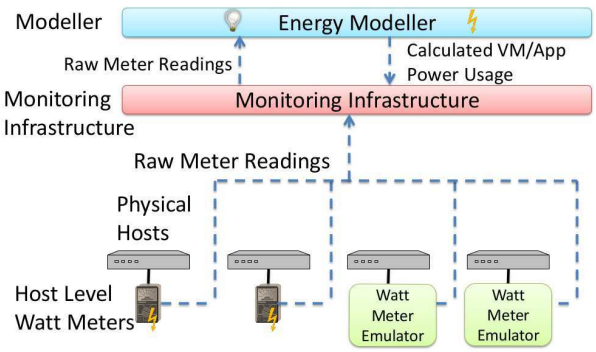


FIGURE 4 Monitoring architecture for the Energy Modeller

The Energy Modeller in summary has three main roles. The first is at deployment time when workload is being placed upon the physical infrastructure. In Clouds the VM manager can utilise power consumption predictions for the placement of VMs[51]. Similarly in HPC environments RJMS such as SLURM[3] may also use power consumption predictions to place jobs. The second is at operation time when the environment is monitored and this information is utilised to aid adaptation. The third covers the aspect of billing and monitoring, ensuring energy usage can be monitored and potentially charged for. In each case the Energy Modeller is required to attribute power consumption to applications or virtual machines. This allocation may include existing VMs/Applications and those that are scheduled to be deployed.

The major components for energy monitoring with the energy modeller are shown in Figure 4. At the lowest level the monitoring utilises various data sources, such as IPMI and RAPL that are integrated into the physical hosts or Watt meters[44] that are attached to the physical host machines. In addition to direct measurement, power consumption can be inferred by the use of a model that translates resource utilisation of a host into power consumption. This process essentially emulates a Watt meter and is shown in Figure 4 as the Watt meter emulator.

The Watt meter emulator is a tool that prevents the need for having Watt meters attached to every physical host in the data centre, thus enabling monitoring at scale. It utilises recent utilisation information and an energy calibration model to decide what the current power consumption of a physical host is. In doing so it removes the requirement for attaching Watt meters to every physical host.

The power consumption as reported from these various data sources is published in monitoring infrastructures, such as Zabbix[15]. These values for host power consumption can then be utilised by the Energy Modeller for the purpose of estimating the power consumption of an application or VM.

5.3 | Power and Energy Estimation

One aspect of energy modelling is the potential to attribute power consumption to "energy users" such as applications and virtual machines. The term energy user is used in place of VM or application and is essentially interchangeable. The Energy Modeller's main role in this case is to assign energy consumption values to the energy user from the values obtained at host level. This is needed because energy consumption associated with energy users is not a directly measurable concept. Rules therefore establish how the host energy consumption is assigned to each energy user. The host energy consumption can be fractioned out in one of several ways, within the Energy Modeller, which is discussed below:

CPU Utilisation Only: This uses CPU utilisation data for each energy user and assigns the energy usage by the ratio produced by the utilisation data. (Available for: Historic, Current, Predictions). This is described in Equation 1 where EU_P_x is the named Energy user's power consumption, Host_P is the measured host power consumption. EU_Util_x is the named energy user's CPU utilisation, EU_Count is the count of Energy users on the host machine. EU_Util_y is the CPU utilisation of a member of the set of energy users on the named host.

$$EU\_P_x = Host\_P \times \frac{EU\_Util_x}{\sum_{y=1}^{EU\_Count} EU\_Util_y} \tag{1}$$

CPU Utilisation and Idle Energy Usage: Idle energy consumption of a host can also be considered. Using training data the idle energy of a host is calculated. This is evenly distributed among the energy users that are running upon the host machine. The remaining energy is then allocated in a similar fashion to the CPU Utilisation only mechanism. (Available for: Historic, Current, Predictions). This is described in Equation 2 where Host_Idle

is the host's measured idle power consumption. This provides an advantage over the first method in that an energy user is more appropriately allocated power consumption values and prevents it from using no power while it is inactive.

$$\text{EU\_P}_x = \text{Host\_Idle} + (\text{Host\_P} - \text{Host\_Idle}) \times \frac{\text{EU\_Util}_x}{\sum_{y=1}^{\text{EU\_Count}} \text{EU\_Util}_y} \tag{2}$$

Evenly Shared: In the case of predictions CPU utilisation is clearly not easy to estimate, thus predicted power consumption can instead be evenly fractioned amongst energy users that are on the host machine. The default for predictions is to share out power consumption evenly as per Equation 3, this is chosen as it relies less upon forecasting individual energy user's workloads and is hence favourable given the potential inaccuracies. A slight variation also exists which counts the CPU cores allocated to each of the Energy users and allocating power based upon this count (Equation 4). Equations 3 and 4 describe this even sharing rules where Host_Predicted is the estimated amount of CPU Utilisation on the host where the energy user resides. This value is derived from an average of the most recent measurements. EU_CPU_x is the amount of virtual CPUs (cloud usecase)/CPUs (HPC usecase) allocated to the named energy user while EU_CPU_y is the amount of virtual CPUs/CPUs allocated to an energy user on the named host.

$$EU\_P_x = Host\_Predicted \times \frac{1}{EU\_Count} \tag{3}$$

$$EU\_P_x = Host\_Predicted \times \frac{EU\_CPU_x}{\sum_{y=1}^{EU\_Count} EU\_CPU_y} \tag{4}$$

The default method chosen on the Energy Modeller is Equation 2 for current and historic values and 3 for predictions. Once the Energy Modeller has assigned energy values to a given energy user it reports these values back to the monitoring infrastructure, thus providing VM and application level power consumption values. The energy modeller remains flexible in its rules for designating power consumption at the level of applications and virtual machines, Equation 4 may be used in cases where the applications running in the environment are heterogeneous with respect to how many cores are utilised.

6 | EVALUATION

The evaluation performed in this section focuses upon the evaluation of the model through the use of the energy modeller and Watt meter emulator. The evaluation covers two distinct parts. The first covers experimentation within a HPC based environment focusing on the applicability of the energy modeller and calibration in such environment. The second focuses on a Cloud based environment considering the calibration of power/energy models that are used to measure power consumption of applications or VMs, thus needing finer temporal resolution.

6.1 | Objectives

The experimentation follows the energy modeller's calibration process which involves inducing load at selected preset values onto a physical host and measuring the power consumption that the load causes. In Section 6.3.1 the focus is upon HPC environments and in particular the comparison and integration of RAPL and IPMI based measurements. The overall aim of this section is to evaluate the accuracy of both source of power measurements and consider how they might be integrated to provide more accurate values.

The second phase of experimentation covers a Cloud based environment (Section 6.3.2), with the aim of exploring the suitability of IPMI's power measuring functionality for determining the power consumption of an application or a virtual machine. In particular the question of how IPMI based calibration data can be processed is considered so that it provides as similar accuracy to a Watt meter as possible is considered.

6.2 | Experimental Setup

Two separate environments are used for the experiments presented here. The first represents a HPC based environment and the second represents Cloud computing environment.

The HPC environment uses SLURM [3,6] for monitoring. It has 48 physical nodes of which three different physical hosts were used for measurement, nd32, nd36 and a GPU based node. nd32 is a bullx B510 double compute blade which has 2 $\times$ Intel Xeon E5-2660 (SandyBridge) 8 cores CPU at 2.6GHz with 4 $\times$ 32GB DDR3-1600 ECC SDRAM RAM and 2 $\times$ 256GB hard disks. nd36 differs in running at 2.2Ghz and is otherwise the same as nd32. The GPU node is the same as nd32 but has two NVidia Tesla K20Xm GPU as well. The theoretical maximum performance of each host is 2,253 Gflops and 2,662 respectively. The CPU frequency scale governor was set to performance for both hosts. They have an InfiniBand

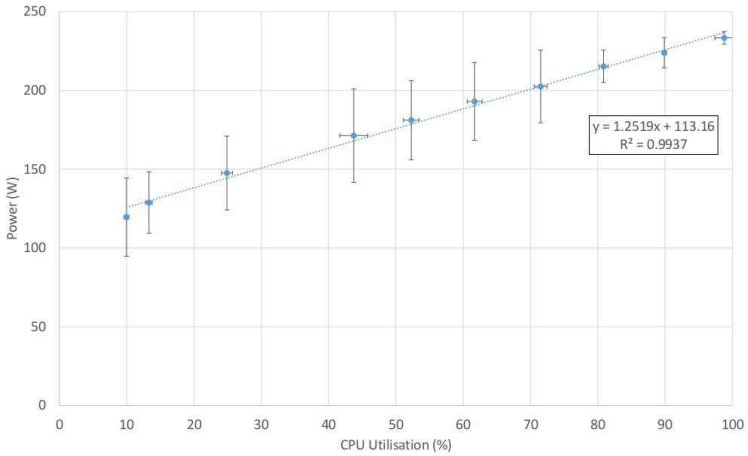


FIGURE 5 Calibration using SLURM and IPMI with incrementing CPU load on node 32

Interconnect (3 × ISR 9024D and 2 × ISR 9024D-M) for high-speed interconnection (20Gb/s) between other nodes in the cluster (i.e. compute, management and I/O nodes). The physical host used the Lustre[52] distributed file system. The sensor data from IPMI and RAPL was taken every 5 seconds and reported through the SLURM monitoring infrastructure. The IPMI sensor in the HPC case provides values in intervals of 7W.

The experimentation in the Cloud environment was performed on a Cloud testbed, that uses Open Nebula 4.10.2[53] and Zabbix 2.4.4[15] for monitoring. The physical host that was measured is a Dell PowerEdge R430 Server commodity server that is monitored through IPMI. The physical host tested has two 2.4GHz Intel Xeon E5-2630 v3 CPUs with 128GB of RAM, a 120GB SSD hard disk and an iDRAC Port Card that is IPMI 2.0 compliant. For the purpose of creating a baseline to compare IPMI based power meter values a WattsUp Meter Pro[44] is attached, with an accuracy of +/- 1.5%. The readings from the Watts Up meter and IPMI sensor were taken and reported to Zabbix every second and every 5 seconds, respectively. In post processing the values reported by IPMI were interpolated, in order to compare data to the Watt meter. The IPMI sensor uses an inbuilt time window of 60 seconds. Zabbix was installed on a separate server as to the host undergoing measurement as to avoid unnecessary additional load. The physical host used network attached storage (NAS) that was used for VM images. This NAS was backed by a PowerEdge R730xd server with an Intel Xeon E5-2603 v3 CPU, with 64GB of RAM, 48TB Hard disk space with an additional 400GB SSDs for caching with a 4Gb/s bonded network connection.

In the experimentation a sequence of loads were applied to each physical host. This load is synthetic in nature, but is more precisely controlled than any other arbitrary benchmarking tool allowing for better quality calibration data to be obtained. The load induced on the physical hosts in both environments ranges from 0% CPU usage up to 100% in increments of 10%. In order to generate this load `Stress`[54] was used, along with `cpulimit` and `taskset`. Each load period lasted 120 seconds with a 0.5 second gap between load periods. The data was then processed so that the start and end of each load period was ignored, allowing for high quality settled values to be used for the calibration data. In the 120 second run for each load period data from 20s-100s is used for calibration. In order to generate full load 16 threads were launched in the HPC case and 32 threads were launched in the Cloud case. The threads were then mapped using `taskset` to the CPU cores on the physical host. `cpulimit` was used to set the intended load to a given utilisation level and each interval of load was induced for 120 seconds. In the Cloud environment in order to represent a realistic setup for the physical host the CPU scheduling governor was set to the default option of on demand and hyper-threading was enabled with all sleep states been available.

6.3 | Results & Discussion

6.3.1 | HPC Testbed

The sequence of calibration as described in Sections 3 and 6.2 was run on a physical host, in the HPC based environment. The average for each fixed CPU utilisation load period is shown in Figure 5, with standard error shown as well. The R^2 value is small at 0.9937, thus demonstrating a good fit for the data.

The model once trained can then be used to estimate power consumption of an application running under normal working conditions. The benchmarking application Hydro[55] is chosen for this purpose. The usage of the model is demonstrated in Figure 6. The trace shows close agreement between the IPMI acquired measurements and the model generated estimated power. The utilisation of a model has two main advantages over using IPMI measurements directly. The first is given the model follows CPU utilisation which can more easily be mapped to an individual process

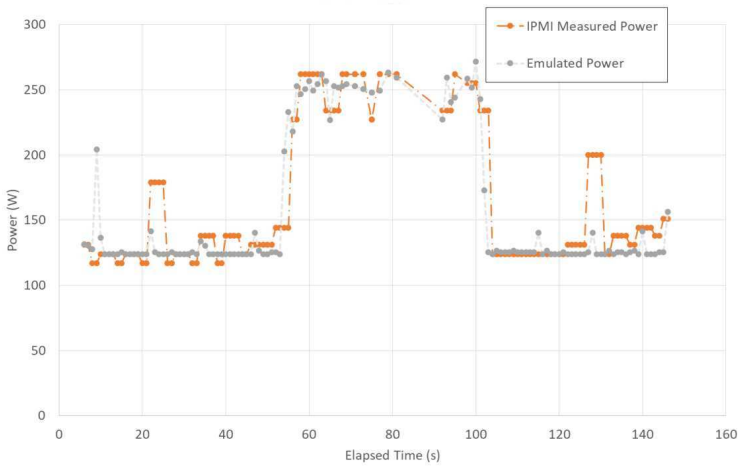


FIGURE 6 A trace comparing IPMI measurements and estimated power using IPMI based calibration data on node 32

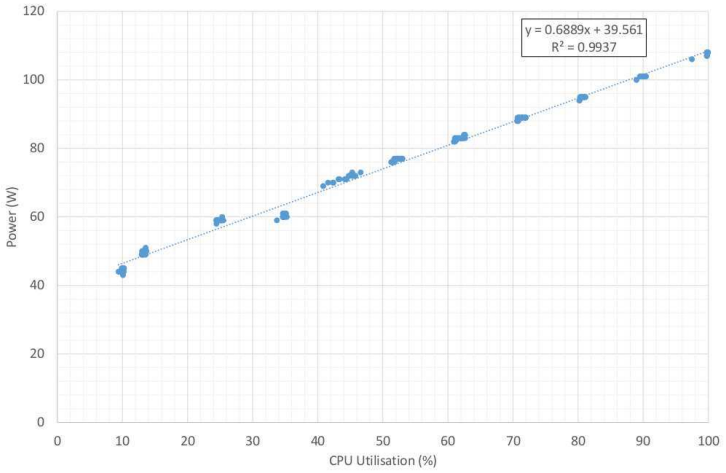


FIGURE 7 Raw Calibration using SLURM and RAPL with incrementing CPU load on node 36

allowing power consumption to be mapped to applications. The second is the model can provide its output faster and without long term averaging as compared to IPMI. More generally models over direct measurement may be used to forecast future power consumption, which has particular use when future workload is predicted and the model can then be used to manage future power consumption.

In Figure 7 the gathering of calibration data using RAPL is illustrated. The power consumption shown to be used is lower than the values gained via IPMI. This is because RAPL has one particular limitation in that it does not report the whole host's power consumption. In the case presented here it measures both the CPUs package and the DRAM power consumption. It thus works well for determining the CPU's power consumption which is a key contributor to the host's active power. IPMI however can report power consumption for the whole host. This is of course advantageous but it suffers from a lower resolution, in both the temporal (by utilising running averages) and power based domains (the testbed in use has a resolution of 7W). The resolution in comparison for RAPL as measured through SLURM on our testbed is 1W. The data shown in Figure 7 has also not undergone the same averaging at each measurement interval. The first 17 seconds of data from fixed CPU utilisation run was ignored, thus ensuring noise was removed, before the CPU utilisation level settles at the target utilisation level.

In comparison RAPL has higher resolution than IPMI, so this is useful, but it also requires the Model Specific Registers (MSR) kernel module to be loaded under Linux, and in order for it to run under Windows it requires administrator privileges. Models in this case do not require any elevated privileges. The IPMI and RAPL values do not exactly correlate and in particular RAPL does not measure all power consumption of the physical host. The calibration data from both datasets can however be used in such a way that the calibration data can be merged.

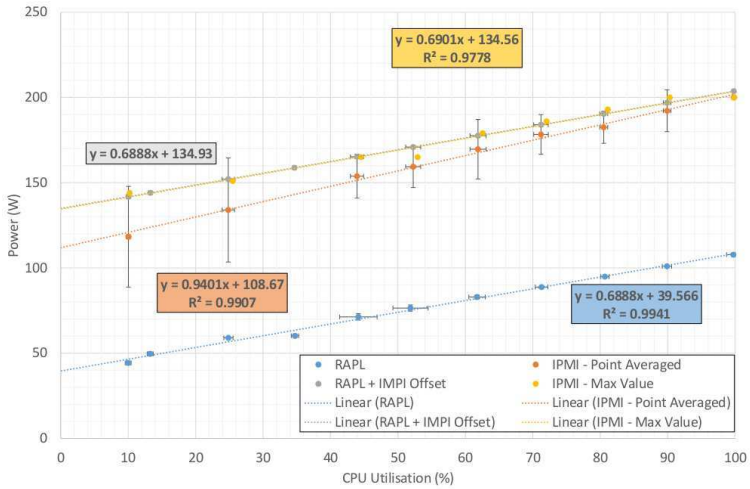


FIGURE 8 Calibration data using SLURM with RAPL and IPMI with incrementing CPU load on node 36

The first element of this is the intercept, which accounts for the idle power consumption of the physical host, IPMI better represents this value, even if it has not got the highest resolution. RAPL however has higher resolution so is better at determining the change in power consumption given a set CPU utilisation level. The result of merging the two datasets properties is shown in Figure 8. The plot shows, RAPL, RAPL + IPMI offset, IPMI using point averaging at each fixed utilisation level and IPMI with the maximum power value shown for each CPU utilisation level. In Figure 8 RAPL is clearly too low to represent the host's full power consumption, but to its advantage the sensor data includes both the CPU and the DRAM power consumption which are the principle sources of change in power consumption for the physical host being measured. IPMI shows the full power consumption used, but using this alone risks granularity of data issues. IPMI using averaging seems a reasonable approach, especially in regards to the highest power consumption which shows very low variance in the values obtained. The gradient of RAPL and the averaged IPMI data however do not correlate. It could be expected given limited disk utilisation and no graphic card utilisation that this was more likely to occur. Each measurement from IPMI at a given utilisation level will however give a small spread of measured values. Taking the maximum value corrects the gradient, such that it matches RAPL. This can be done for several reasons, with caution. The maximum value derives from a single value instead of the entire series which is then averaged. The maximum value is therefore subject to a greater chance of error as compared to the average, but if the variance of the entire series is low it does not suffer from averaging issues that will artificially lower the datapoints power consumption. The calibration process forces a process which would otherwise take 100% utilisation to only use a portion of the overall CPUs available resources. The capping process using `cpulimit` therefore creates an artificial ceiling to which the utilisation and consequentially the power can achieve. Thus the closer to this ceiling the datapoint represents, the more reliable the result can be considered to be, given other values are likely to under represent the power consumed. This is particularly the case when quantisation as used in IPMI can under report the power consumed.

Figure 9 like Figure 6 considers a trace and the accuracy of the estimated power consumption as compared to the actual measured value. In this case the estimate derives from the calibration data obtained using both RAPL and IPMI as shown in Figure 8. The trace shows a good fit overall with its Root Mean Square Error (RMSE) of 17.02 and an average deviation between IPMI measured and the model of 4.48% and an absolute deviation of 10.41%. Thus showing that although any individual power measurement may be subject to some error over the longer term the energy measurement remains more accurate. It should be noted that the deviation is in part due to IPMI's resolution of 7W and that the model driven power estimates do not have such discrete fixed values as seen in the IPMI based measurements. This is demonstrated in applying quantisation to the estimated power values such that they only express values shown by IPMI. This quantisation causes the error to drop to 1.74% and the absolute error to 8.48%. In the next section comparisons are drawn against a Watt meter, which is a better illustration of the model and calibration processes' accuracy.

The final experiment (Figure 10) within the HPC testbed shows the use of the proposed modelling strategy within the context of accelerators such as GPUs. The average error is -1.38W within the trace representing -1.13% of the host's idle power consumption. This average error is important when reporting power consumption issues. In considering longer periods of time and in particular energy consumption the average absolute error is 6.23W representing 5.08% of the host's idle power consumption. Most of this error is derived from transition periods between no load and full load, leaving the median absolute error 2.45W or 2.00% of the idle host's power consumption, so the model remains particularly reasonably accurate, given the usage of a simple model with the focus on the quality of the calibration data.

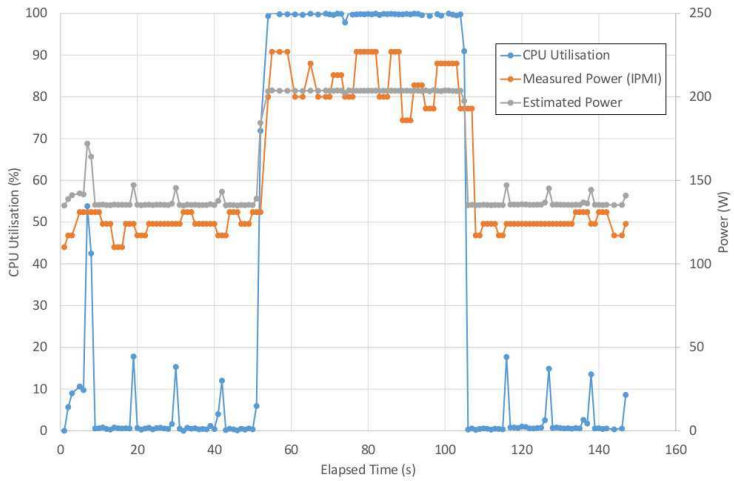


FIGURE 9 A trace comparing IPMI measurements and estimated power using IPMI+RAPL based calibration data on node 36

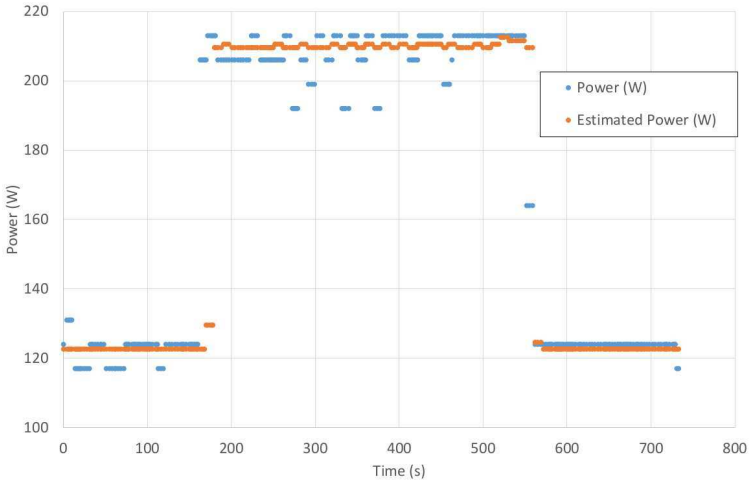


FIGURE 10 A trace showing the estimated power using a bi-modal strategy for GPUs

6.3.2 | Cloud Testbed

In this section calibration work on a Cloud testbed is discussed. In this case a Watt meter and IPMI measurements are utilised. The focus is upon using IPMI measurements in such a way that it closely as possible matches the gold standard of using an attached Watt meter, who's use is impractical in large data centres.

In Figure 11 the overall trace of the calibration run is shown. It shows multiple measurements for each set CPU utilisation level been gathered via IPMI and the Watt meter along with the CPU load induced on the physical host. The Watt meter at the start of some periods of induced load especially at 10% and 20% CPU load shows spikes, before the load settles. This is in contrast to the IPMI sensor that is unable to detect any change in power consumption at 10% CPU load. This is due to the granularity of the sensor. It exhibits only 9 distinct values bands within the measurement range used (112W - 224W in 14W increments). The initial measured idle is 117W while at 10% load it is 124W and with only 7W difference this is undetectable using IPMI.

IPMI undergoes averaging, which results in the peak associated with IPMI been offset to the right of the Watt meter's reported values. This suggests that if accurate calibration is desired that these values should only be used after the average window has passed while sustained consistent load is in effect. The IPMI power values also under report the power consumption by seemingly only rounding down towards the last permissible increment.

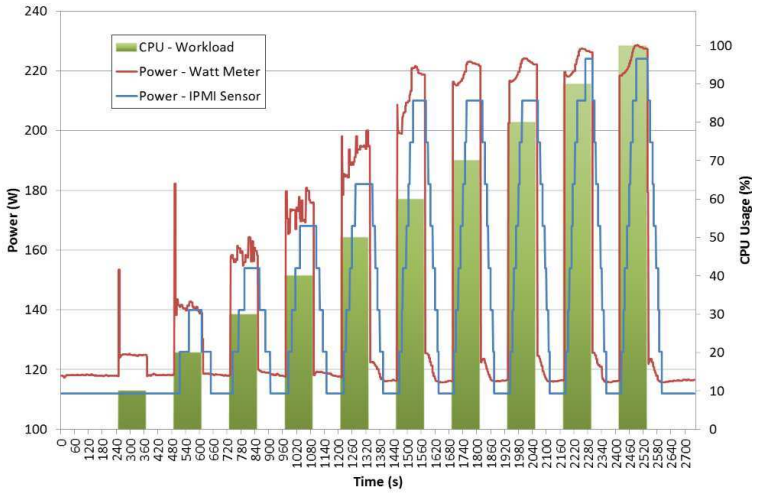


FIGURE 11 Trace of IPMI and Watt meter measurements with incrementing CPU load

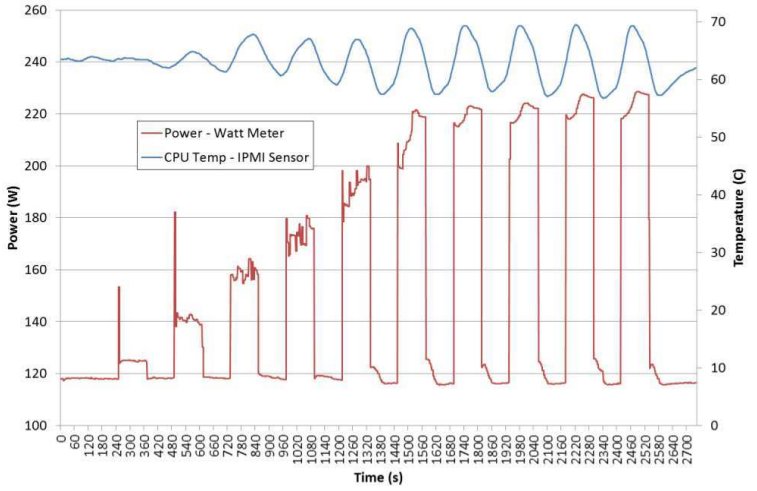


FIGURE 12 Trace of Watt meter and temperature measurements with incrementing CPU load

At 60% CPU utilisation and above it can be noticed that the system's power consumption becomes capped at around 228W, after this point it is speculated that the CPU is throttled to meet it's Thermal Design Power (TDP) budget. It can therefore be seen that a purely linear model as seen in much of the literature does not apply in the context of our machine, as also seen in [56].

In Figure 12, the effect of temperature measured by IPMI on the power consumption can be examined. It shows a higher than expected variance in power during the sustained 120 second workload. The correlation between CPU load and CPU temperature can clearly be seen. The temperature at the start of our experiment before any load is induced starts at 63°C, yet lowers to 53°C at the lowest point during our experiment, which occurs soon after a load period has completed and is a result of the fans cooling the CPU past its normal idle temperature. At 50% CPU utilisation and above in our test setup, the power consumption as reported by the Watt meter shows an initial slope and then a tail in which the power consumption doesn't immediately drop down to idle once the load has finished. It is speculated that this is the effect of Ohm's law and the increased resistance caused by the higher operating temperature of the CPU. It has previously been shown in Wang et. al. [57] that a non-linear relationship between the leakage power and temperature exist, making this assumption reasonable. In addition power consumption induced by the fans as part of the increased requirement for cooling, could be considered. Thus as the CPU further heats at the start of a load period an initial slope is created due to heating and the increase in fan speed. The power consumption stabilises and then at the end of the load period drops, yet the remaining additional heat takes time to dissipate, thus causing the tail.

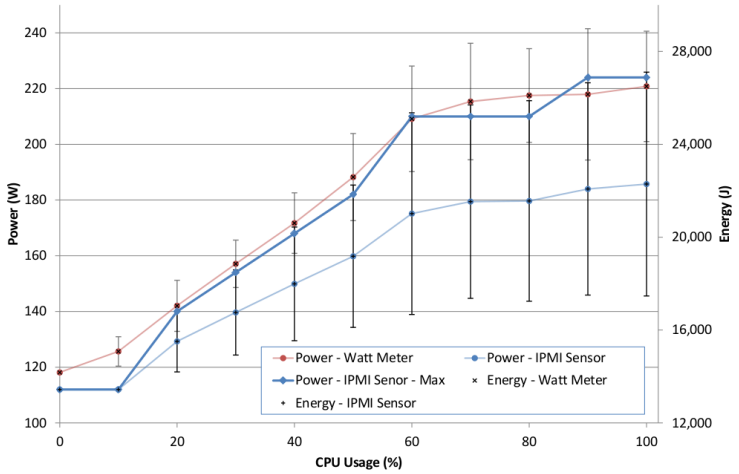


FIGURE 13 CPU load vs power and energy consumption

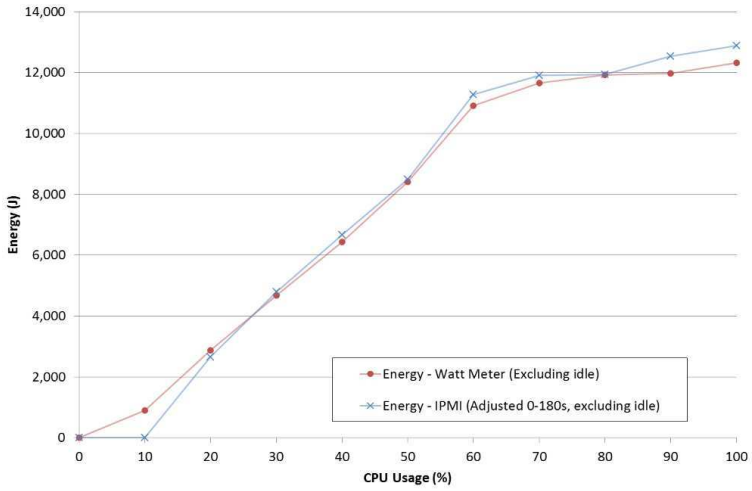


FIGURE 14 CPU load vs energy consumption adjusting to compensate for idle host power consumption

In Figure 13 the CPU load and power consumption calibration data is processed from the raw data (shown in Figure 11) where the data points over the 120 seconds of each workload are utilised. The aim of this analysis is to collect data that can be used in estimating power consumption from CPU load. To do this at each specific CPU utilisation level the values are averaged and 95% confidence intervals are shown or in the case of IPMI a maximum power value is shown as well. It is seen that IPMI consistently under reports the power consumption and also the overall energy consumed. This is reduced by using the IPMI maximum value instead. It can be seen that the error in averaged IPMI values are also larger particularly when the CPU load is higher. This error is due to the averaging window that the IPMI device is using when taking measurements. The maximum value works because it discards the initial values where the averaging window is yet to only represent a time period under consistent fixed load. The max value also favours IPMI values from the highest quantisation level possible for a given workload, while the average favours the lower quantisation level. The maximum value in this case clearly shows the effect of the quantisation levels in the IPMI's measurement. The effect of the stepped increments can be seen in Figure 11 in how at 10% CPU utilisation that IPMI values do not register the change in power consumption. It can however be seen that each data point in the series, although offset, follows the same trend.

Figure 14 shows the effect of making two adjustments, that means calibration data obtained by IPMI more closely matches the data obtained from the Watt meter. Firstly the idle power consumption of the server is removed thus only the additional energy consumption of the application is considered and secondly the window size is increased for the IPMI measurements from 120 seconds to 180 seconds. This takes account of the entire averaging window used by IPMI which is fixed at 60 seconds. After these changes it can be seen that the two lines nearly directly correlate, with the Watt meter and the IPMI sensor closely agreeing in the range 20-80 % CPU utilisation but with slightly more error at the high and low

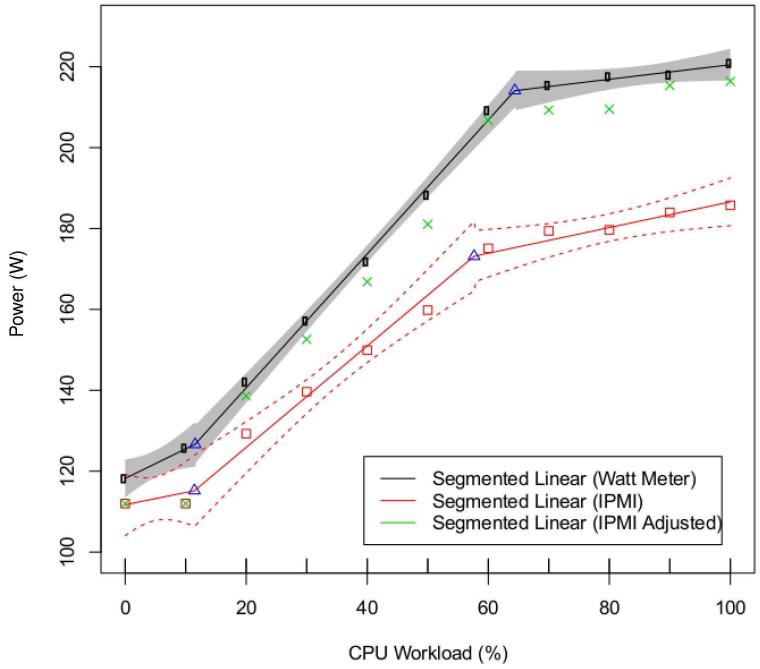


FIGURE 15 CPU load vs power and energy consumption - Compensating for inaccuracies in IPMI measured values

	Multiple R^2	Adjusted R^2
Watt Meter Segmented	0.9989	0.978
Watt Meter Linear	0.9358	0.9287
IPMI Segmented	0.9946	0.9891
IPMI Linear	0.9417	0.9352
IPMI Adjusted Segmented	0.9928	0.9857
IPMI Adjusted Linear	0.9285	0.9206

TABLE 1 The fit data for both linear regression and segmented linear regression

ends. The application of these two simple rules thus illustrates how IPMI can be used to produce a similar result to an actual Watt meter, albeit for the energy consumption of a physical host, VM or application. This processing of the data additionally illustrates how IPMI correctly represents the change in energy consumption, even though the temporal granularity/averaging spreads the measured energy value across a period of time.

To derive the current power consumption of an application from the model is more useful than its energy consumption alone. Figure 15 demonstrates how this can be achieved. It shows a graph of calibration data for power consumption vs CPU utilisation along with confidence intervals of 95% for the Watt meter and IPMI results. The adjusted IPMI confidence intervals are very similar and thus excluded to avoid overly filling the graph. The fit was generated in R using segmented linear regression. Figure 15 additionally shows IPMI gathered data after adjustments. It can be seen how IPMI without processing under reports the power consumption and that the correct answer is reported by the Watt meter. IPMI can be used to get a closer answer to the Watt meter by ignoring the first 60 seconds of datapoints. This works as the averaging window used by IPMI will no longer reflect a period of time before the load was induced and measurements will only reflect the CPU at the load specified. Once this is done the IPMI calibration line fits much more closely to the Watt meter's line. This means in the context of calibration that the load should be induced for at least the length of the averaging window, in order to get a decent calibration. The R^2 values for the fitted lines are shown in Table 1. Once this model has been constructed using the IPMI data, CPU counters can then be used in conjunction with the model generated in order to get rapid and accurate values for the power consumption. Using the maximum value instead of averaging each point provides a similar fit to the adjusted IPMI, although is subject to some error at detecting the breakpoints in the segments, thus more runs at different CPU utilisation levels are required to correct detect such breakpoints positions accurately. The aim would be to find the CPU utilisation level where it is true that the measured power value just changed to the next increment in the IPMI's measured power value.

	WM	IPMI	IPMI-adj
Average error (W)	-0.20	-18.35	-6.50
Average absolute error (W)	15.68	21.88	18.92
Average error/idle power	-0.17%	-15.75%	-5.58%
Absolute average error/idle power	13.46%	18.78%	16.24%

TABLE 2 Error between Watt meter reading and the model generated estimate of power consumption

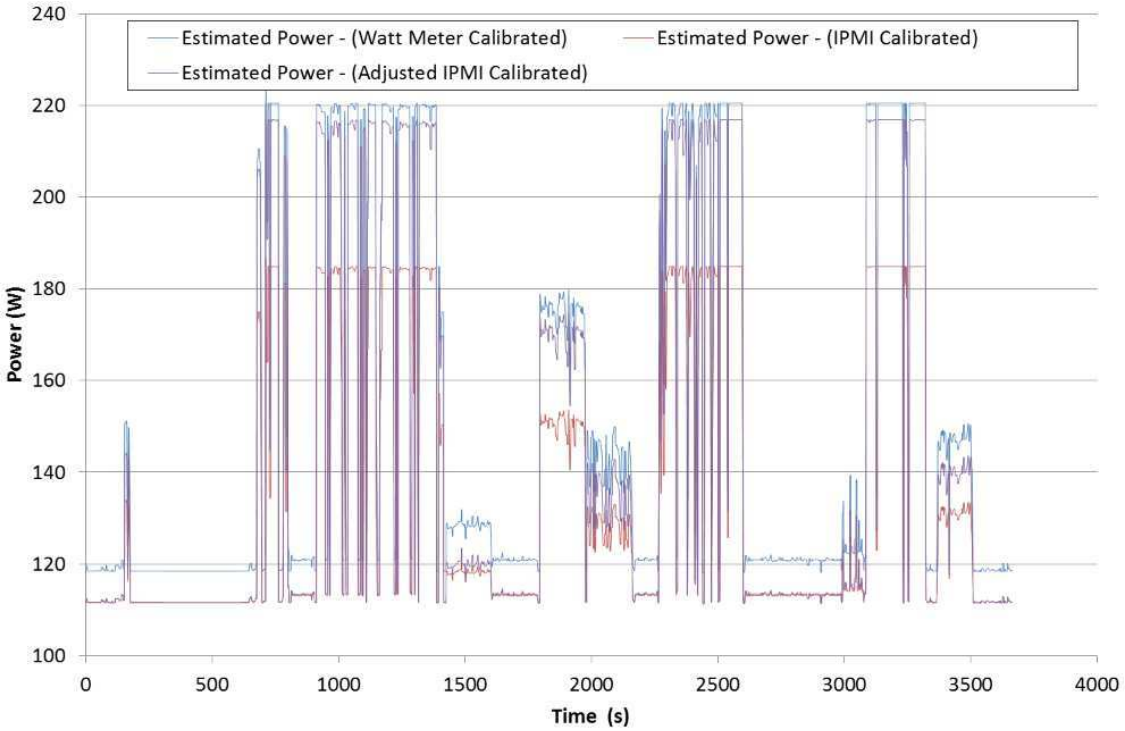


FIGURE 16 Trace of a workload induced by the Phoronix testsuite (CPU)

The remaining focus of this section, is to access the validity of the changes made to the calibration data in the context of IPMI through analysing the accuracy of the power and energy predictions made from a less synthetic workload. A VM was created on the host with 32GB RAM and 32 virtual cores. This gives the VM the possibility of using all physical cores of the host machine. The Phoronix testsuite [58] was then used as a means of inducing a workload. The benchmarking suite then runs for an hour inducing load on the system, with the resultant trace shown in Figure 16. Figure 16 shows the use of the Watt meter emulator with the results from three different calibration datasets. These datasets having been gathered via a Watt meter, by IPMI and via IPMI with the same adjustments as used in Figure 15. It clearly shows how the estimated power consumption for the adjusted IPMI more closely matches the Watt meter generated calibration data's trace. The average error and absolute average error for this trace is shown in Table 2, for Watt Meter calibrated (WM), IPMI and adjusted IPMI (IPMI-adj).

It can be seen in terms of estimating energy consumption of an application the adjustments made to IPMI have made a substantial improvement to the average error (11.85W or 10.17%). Thus over time the estimation of energy consumption will be far more accurate. In considering the absolute error it can be seen while the model used to estimate the actual power consumption has errors a reduction in the error from IPMI alone is also realised (2.96W or 2.54%). This demonstrates how a single power value may have inaccuracies but for the overall energy consumption it will eventually converge to the real value in the context of this workload. The difference in error between IPMI and the Watt meter remains, principally as a result of the lack of resolution of the IPMI based power sensors, having eliminated averaging issues during the calibration run. This can only be resolved by hardware vendor based improvements of these power sensors. Until this improvement is realized, this leaves our models and careful calibration as the only solution for gaining reliable estimates of current power consumption. Models such as ours will retain their usefulness for the prediction of future power consumption of a given workload.

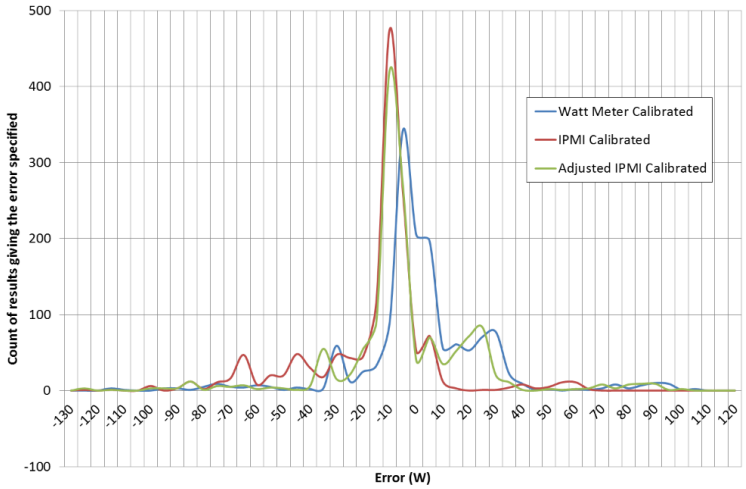


FIGURE 17 A distribution of errors in the model's accuracy as compared to the power meter reading

Finally Figure 17 illustrates the errors associated with the trace. The deviation from the actual power consumption for each estimated power value is calculated and shown on the x axis, while on the y axis the count of how many estimates with that error are shown. Therefore the more estimates that are close to zero Watts of error the better the model will predict the power consumption. In addition the prediction of the energy consumption will be better if the error distribution is more symmetric. Figure 17 shows how the IPMI based calibration data biggest peak has a slight offset from 0 underestimating the power consumed. The adjusted IPMI makes an improvement on this with a peak centred closer to 0W of error. The Watt meter based calibration performed best in that its average error was -0.20W. Aside from observing the proximity to the ideal of centring around zero Watts of error, other errors are shown. These tend to result from transition periods between distinctly different levels of CPU load and timing issues between the different types of metric values being gathered given that measurements were gathered in a distributed system.

7 | LESSONS LEARNED AND BEST PRACTICES

In both HPC and Cloud based environments the measurement of Host power consumption is unsurprisingly similar. In the Cloud testbed, instrumentation with Watt meters was more practical than in the HPC environment utilised. At scale neither scenario lends itself to direct measurement with Watt meters. IPMI and RAPL however offer an alternative approach which can be utilised together to give a reasonable approximation of the true measurement. The resolution and the limits to the context of measurement to which these alternative measurements are taken gives rise to considerations in how the measurements are taken. The utilisation of models enables future predictions in power consumption to be made.

The work here supports[9] in that energy measurements over a sustained period are likely to be statistically correct, while individual power measurements contributing to the total energy measurement may be subject to some error. In regards to measurement over short periods of time this work differs in that it utilises models. The models are calibrated over long well structured experimental runs thus measurements over short periods of time are only likely to be subject to errors in the model, rather than measurement error and synchronizing application start and finish times to load and measured power.

The synchronisation of power and load measurements is important, but the criticality of this differs at runtime as compared to calibration time. During calibration the synchronisation of measurement values is less important and a focus upon ensuring that the measured values represent the physical machine in the same running state is more important. This gives rise to considering how the measurements are treated, such as any running averages that are being undertaken. Such mechanisms give rise to a focus on instantaneous values that report the current state on the spot, rather than reporting against a period of time. It also in structured (calibration) loads, worth considering utilising the maximum value obtained, for power consumption, given that it is a way to eliminate the effects of averaging in fixed load environments. Load periods during calibration can also be considered as a means to eliminate measurement error. Waiting for a stable state to present itself and discarding earlier measured values not only avoids averaging issues during structured load but also eliminates noise caused during load initialisation when the load capping feature has not fully taken effect. Measurements during gathering can also be delayed based upon network location. This means various strategies should be employed such as ensuring measurements have meta-data such as timestamps or ensuring calibration is done locally when a common timestamp and relatively short delay in recording values can be used, which avoids synchronisation issues between different types of metric.

In contrast to calibration time, averaging recent utilisation data at operation time can in some cases be useful. Using a longer time window for a measurement can generate a smoothing effect on the data at the cost of responsiveness and overall accuracy. This avoids rapid fluctuating estimated power consumption values upon which decisions may be based, can give more consistent decisions. IPMI with granularity issues can suffer from rapid changes between values due to is ability to resolve different values for power measurement.

Both Clouds and HPC environments are becoming ever more heterogeneous, with the increasing use of accelerators. Clouds given virtulization have additional overheads in that accelerators need either assigning to virtual machine instances or correctly virtulizing to enable sharing, thus changing the model by which power consumption can be attributed to a given application or virtual machine. The work performed here focuses upon CPU utilisation, suggests that the general techniques used can be applied to other power consuming domains such as GPUs and other accelerators inside Cloud and HPC environments.

As stated in Section 2, Kwapi [41] is the most closely related work. The focus in this paper is to extend this work to VM and application level power consumption which increases the user's awareness of power consumption. This is especially useful in the context of Smart Grids, where power consumption can be shaped to meet a given power profile. Heterogeneity in such environments offers both additional complexities but also opportunities to better shape power consumption to the required demand profile.

8 | CONCLUSION AND FUTURE WORK

In this paper both IPMI, RAPL and Watt meters have been used as a means of generating models that can be used inside Cloud and HPC environments for the purpose of estimating power consumption of both the physical hosts and virtual machines. Both IPMI and RAPL have attractive properties regarding the measurement of power and energy consumption but neither match the quality of a Watt meter. RAPL for example shows the change in power consumption well but underestimates the whole system power consumption while IPMI has a lower resolution. IPMI although having this relatively low resolution can be used in various specialised scenarios. These include showing the energy consumption due to additional load of an application if datapoints after the load has ended are taken into account, due to the effect of an averaging window used by the IPMI device. IPMI can be further used as part of calibration of a host's power model if calibration runs with a continuous load takes longer than the averaging window, with the initial datapoints being discounted. This gives rise to the possibility of calibrating power models for large data centres, even though the IPMI measurement equipment has not achieved a high level of accuracy. These power models thus serve two purposes. The first is that they can be used to predict future power consumption, by estimating workload. The second is that they can be used to make more rapid estimates of power than the readily available measurement equipment allows, given the access to the faster more accurate measurements of CPU load. Aside from IPMI's accuracy, RAPL and IPMI measurements have the prospect of being utilised together, so as to form calibration data, that has both the contributions of the refined resolution of RAPL along with IPMI's more holistic measurements. To refine the accuracy of the calibration proposed here in an automated fashion, a search is expected be performed that finds the CPU load that causes a transition in the IPMI's reported power consumption. This will therefore better cope with IPMIs quantisation effects, which is considered particularly useful future work in cases where purely linear models do not necessarily apply. In addition to this, techniques proposed in this paper will be applied to more heterogeneous environments in which multiple different sources are likely to contribute to the change in power consumption of a physical host.

ACKNOWLEDGMENTS

This work has received support through the EU Horizon 2020 Research and Innovation Programme, under grant agreement 687584 as part of the Transparent heterogeneous hardware Architecture deployment for eNergy Gain in Operation (TANGO) project.

Author contributions

This is an author contribution text. This is an author contribution text. This is an author contribution text. This is an author contribution text. This is an author contribution text.

Financial disclosure

None reported.

Conflict of interest

The authors declare no potential conflict of interests.

SUPPORTING INFORMATION

The following supporting information is available as part of the online article:

References

1. OpenNebula Project . OpenNebula Homepage http://opennebula.org/[17th February 2017]; 2017.
2. OpenStack Foundation . OpenStack Open Source Cloud Computing Software https://www.openstack.org/[15th March 2017]; 2017.
3. Yoo Andy B, Jette Morris A, Grondona Mark. SLURM: Simple Linux Utility for Resource Management. In: Feitelson Dror, Rudolph Larry, Schwiegelshohn Uwe, eds. *Job Scheduling Strategies for Parallel Processing: 9th International Workshop, JSSPP 2003, Seattle, WA, USA, June 24, 2003. Revised Paper*, :44–60Springer Berlin Heidelberg; 2003; Berlin, Heidelberg.
4. Intel . Intelligent Platform Management Interface (IPMI) Specification, V2.0, Rev. 1.1 http://www.intel.co.uk/content/www/uk/en/servers/ipmi/ipmi-second-gen-interface-spec-v2-rev1-1.html[15th February 2017]; 2013.
5. Rotem E, Naveh A, Ananthakrishnan A, Weissmann E, Rajwan D. Power-Management Architecture of the Intel Microarchitecture Code-Named Sandy Bridge. *IEEE Micro.* 2012;32(2):20–27.
6. Georgiou Yiannis, Cadeau Thomas, Glesser David, Auble Danny, Jette Morris, Hautreux Matthieu. Energy Accounting and Control with SLURM Resource and Job Management System. In: Chatterjee Mainak, Cao Jian-nong, Kothapalli Kishore, Rajsbaum Sergio, eds. *15th International Conference on Distributed Computing and Networking: ICDCN 2014*, :96–118Springer Berlin Heidelberg; 2014; Coimbatore, India.
7. Audet J, A Amelkin, C Hebert, et al. IPMI Tool https://sourceforge.net/projects/ipmitool/[15th February 2017]; 2016.
8. Free IPMI Team . GNU FreeIPMI https://www.gnu.org/software/freeipmi/[15th February 2017]; 2016.
9. Hackenberg D, Ilsche T, Schone R, Molka D, Schmidt M, Nagel W E. Power measurement techniques on standard compute nodes: A quantitative comparison. In: ISPASS:194–204; 2013.
10. Desrochers Spencer, Paradis Chad, Weaver Vincent M. A Validation of DRAM RAPL Power Measurements. In: MEMSYS '16:455–470ACM; 2016; New York, NY, USA.
11. Intel . *Intel 64 and IA-32 Architectures Software Developers Manual Combined Volumes: 1, 2A, 2B, 2C, 2D, 3A, 3B, 3C, 3D and 4*. 2018.
12. Hähnel Marcus, Döbel Björn, Völp Marcus, Härtig Hermann. Measuring Energy Consumption for Short Code Paths Using RAPL. *SIGMETRICS Perform. Eval. Rev.*. 2012;40(3):13–17.
13. McCraw H, Ralph J, Danalis A, Dongarra J. Power monitoring with PAPI for extreme scale architectures and dataflow-based programming models. In: CLUSTER:385–391; 2014.
14. Wilke Claas, Götz Sebastian, Richly Sebastian. JouleUnit: A Generic Framework for Software Energy Profiling and Testing. In: GIBSE '13:9–14ACM; 2013; New York, NY, USA.
15. ZABBIX SIA . Homepage of Zabbix:: An Enterprise-Class Open Source Distributed Monitoring Solution http://www.zabbix.com/[08th August 2014]; 2014.
16. Fatema Kaniz, Emeakaroha Vincent C, Healy Philip D, Morrison John P, Lynn Theo. A survey of Cloud monitoring tools: Taxonomy, capabilities and objectives. *Journal of Parallel and Distributed Computing*. 2014;74(10):2918–2933.
17. GEMBIRD Deutschland GmbH . EGM-PWM-LAN data sheet http://gmb.nl/Repository/6736/EGM-PWM-LAN_manual---7f3db9f9-65f1-4508-a986-90915709e544.pdf[15th January 2015]; 2013.

18. Kansal Aman, Zhao Feng, Liu Jie, Kothari Nupur, Bhattacharya Arka A. Virtual Machine Power Metering and Provisioning. In: SoCC '10:39–50ACM; 2010; New York, NY, USA.

19. Castañé Gabriel G, Núñez Alberto, Llopis Pablo, Carretero Jesús. E-mc2: A formal framework for energy modelling in cloud computing. *Simulation Modelling Practice and Theory.* 2013;39(0):56–75.

20. Laros J H, Pokorny P, DeBonis D. PowerInsight - A commodity power measurement capability. In: IGCC:1–6; 2013.

21. Ge R, Feng X, Song S, Chang H C, Li D, Cameron K W. PowerPack: Energy Profiling and Analysis of High-Performance Systems and Applications. *IEEE Transactions on Parallel and Distributed Systems.* 2010;21(5):658–671.

22. Bohra A E H, Chaudhary V. VMeter: Power modelling for virtualized clouds. In: IPDPSW:1–8; 2010.

23. Yang Hailong, Zhao Qi, Luan Zhongzhi, Qian Depei. iMeter: An integrated {VM} power model based on performance profiling. *Future Generation Computer Systems.* 2014;36(0):267–286.

24. Smith J W, Khajeh-Hosseini A, Ward J S, Sommerville I. CloudMonitor: Profiling Power Usage. In: CLOUD:947–948; 2012.

25. Farahnakian F, Liljeberg P, Plosila J. LiRCUP: Linear Regression Based CPU Usage Prediction Algorithm for Live Migration of Virtual Machines in Data Centers. In: SEAA:357–364; 2013.

26. Schubert S, Kostic D, Zwaenepoel W, Shin K G. *Profiling Software for Energy Consumption.* 2012.

27. Jiang Zhixiong, Lu Chunyang, Cai Yushan. VPower: Metering power consumption of VM. *2013 IEEE 4th International Conference on Software Engineering and Service Science.* 2013;:483–486.

28. Khan Arijit, Xifeng Yan , Shu Tao , Anerousis N. Workload characterization and prediction in the cloud: A multiple time series approach. In: :1287–1294IEEE; 2012.

29. Alzamil Ibrahim, Djemame Karim. Energy Prediction for Cloud Workload Patterns. In: Altmann Jorn, ed. *13th International Conference on Economics of Grids, Clouds, Systems and Services (GECON'2016)*, Springer; 2016; Athens, Greece.

30. Roy N, Dubey A, Gokhale A. *Efficient Autoscaling in the Cloud Using Predictive Models for Workload Forecasting.* 2011.

31. Islam Sadeka, Keung Jacky, Lee Kevin, Liu Anna. Empirical prediction models for adaptive resource provisioning in the cloud. *Future Generation Computer Systems.* 2012;28(1):155–162.

32. Quiroz Andres, Kim Hyunjoo, Parashar Manish, Gnanasambandam Nathan, Sharma Naveen. Towards autonomic workload provisioning for enterprise Grids and clouds. In: GRID:50–57IEEE; 2009; Banff, Alberta, Canada.

33. Djemame K, Kavanagh R, Armstrong D. Energy Efficiency Support through Intra-Layer Cloud Stack Adaptation. In: Altmann J, ed. *13th International Conference on Economics of Grids, Clouds, Systems and Services (GECON'2016)*, Springer; 2016; Athens, Greece.

34. Prevost John J, Nagothu Kranthimanoj, Jamshidi Mo, Kelley Brian. Optimal calculation overhead for energy efficient cloud workload prediction. In: :741–747IEEE; 2014; Waikoloa Village, Hawaii.

35. Adhinarayanan V, Subramaniam B, Feng W C. Online Power Estimation of Graphics Processing Units. In: CCGrid:245–254; 2016.

36. Juurlink B., Lucas J., Mammeri N., et al. Enabling GPU software developers to optimize their applications The LPGPU2 approach. In: DASIP:1-6; 2017.

37. Song S., Su C., Rountree B., Cameron K. W.. A Simplified and Accurate Model of Power-Performance Efficiency on Emergent GPU Architectures. In: IPDPS:673-686; 2013.

38. Leng Jingwen, Hetherington Tayler, ElTantawy Ahmed, et al. GPUWattch: Enabling Energy Optimizations in GPGPUs. *SIGARCH Comput. Archit. News.* 2013;41(3):487–498.

39. Sundriyal Vaibhav, Sosonkina Masha. Joint frequency scaling of processor and DRAM. *The Journal of Supercomputing.* 2016;72(4):1549–1569.

40. Mei X., Chu X., Liu H., Leung Y. W., Li Z.. Energy efficient real-time task scheduling on CPU-GPU hybrid clusters. In: INFOCOM:1-9; 2017.

41. Rossigneux Francois, Lefevre Laurent, Gelas Jean-Patrick, Dias De Assuncao Marcos. A Generic and Extensible Framework for Monitoring Energy Consumption of OpenStack Clouds. In: Sustaincom; 2014; Sydney, Australia.

42. Khan Kashif Nizam, Hirki Mikael, Niemi Tapio, Nurminen Jukka K., Ou Zhonghong. RAPL in Action: Experiences in Using RAPL for Power Measurements. *ACM Trans. Model. Perform. Eval. Comput. Syst.*. 2018;3(2):9:1–9:26.

43. Ganglia Project . Ganglia Monitoring System http://ganglia.sourceforge.net/[13th April 2014]; 2012.

44. Electronic Education Devices . WattsUp? Watt Meters https://www.wattsupmeters.com/secure/index.php[6th August 2014]; .

45. Hackenberg D, Ilsche T, Schuchart J, et al. HDEEM: High Definition Energy Efficiency Monitoring. In: E2SC:1–10; 2014.

46. Kavanagh Richard, Armstrong Django, Djemame Karim. Accuracy of Energy Model Calibration with IPMI. In: CLOUD:648–655IEEE; 2016; San Francisco, California.

47. Dell Inc . Using the BMC Management Utility http://support.dell.com/support/systemsinfo/document.aspx?s={%}0Aslg{&}file=/software/smbmcmu/1.2/en/ug/bmcugc0d.htm[03rd March 2017]; 2014.

48. NVidia Corporation . *NVML API Pages - For GPU Utilization.* 2017.

49. Djemame Karim, Armstrong Django, Kavanagh Richard E., et al. Energy Efficiency Embedded Service Lifecycle: Towards an Energy Efficient Cloud Computing Architecture. In: :1–6CEUR Workshop Proceedings; 2014; Stockholm, Sweden.

50. Kavanagh Richard, Armstrong Django, Djemame Karim, Sommacampagna Davide, Blasi Lorenzo. Towards an Energy-Aware Cloud Architecture for Smart Grids. In: Altmann Jorn, Silaghi Gheorghe Cosmin, Rana Omer F., eds. *12th International Conference on Economics of Grids, Clouds, Systems, and Services*, :190–204Springer International Publishing; 2016; Cluj-Napoca, Romania.

51. Djemame K, Bosch R, Kavanagh R, et al. PaaS-IaaS Inter-Layer Adaptation in an Energy-Aware Cloud Environment. *IEEE Transactions on Sustainable Computing*. 2017;2(2):127–139.

52. Open Scalable File Systems Inc . Lustre http://lustre.org/[04th April 2017]; 2017.

53. OpenNebula Systems . Open Nebula: Flexible Enterprise Cloud Made Simple http://opennebula.org/[17th February 2016]; 2016.

54. Waterland Amos. Stress Project http://people.seas.harvard.edu/{~}apw/stress/[07th March 2016]; 2014.

55. Verdiere Guillaume Colin. HydroBench/Hydro: A 2D Hydro code for benchmarking purposes https://github.com/HydroBench/Hydro[04th April 2017]; 2017.

56. Arjona Aroca Jordi, Chatzipapas Angelos, Fernández Anta Antonio, Mancuso Vincenzo. A Measurement-based Analysis of the Energy Consumption of Data Center Servers. In: e-Energy '14:63–74ACM; 2014; New York, NY, USA.

57. Wang Hai, Wan Jiachun, Tan S. X. ., et al. A fast full-chip static power estimation method. In: ICSICT:241-243; 2016.

58. Phoronix Media . Phoronix Test Suite - Linux Testing & Benchmarking Platform, Automated Testing, Open-Source Benchmarking http://www.phoronix-test-suite.com/[18th February 2016]; 2016.

AUTHOR BIOGRAPHY

photo/richard-photo.jpeg

Richard Kavanagh was awarded a Ph.D. in 2013 and is currently a research fellow at the School of Computing at the University of Leeds. He has experience working in a number of EC-funded projects including OPTIMIS, ASCETiC and TANGO. His research is in the field of Distributed Systems and the complementary paradigms of Grid and Cloud Computing, with a specific interest in quality of service, energy efficiency and resource management.

photo/karim-photo.png

Karim Djemame was awarded a Ph.D. at the University of Glasgow, UK, in 1999, and is currently holding a Chair position at the School of Computing, University of Leeds. He sits on a number of international programme committees for cloud middleware, computer networks and performance evaluation. He was the investigator of various e-Science/Cloud projects including DAME, BROADEN, AssessGrid, ISQoS, STRAPP, OPTIMIS and ASCETiC. He is currently involved in various research projects including TANGO. His main research areas focus on Grid/Cloud computing, including system architectures, resource management, and energy efficiency.