A cloud server energy consumption measurement system for heterogeneous cloud environments

Abstract: With the rapid development of cloud computing technologies and applications in recent years, the number and scale of cloud data centers have rapidly increased while the problem of energy consumption in cloud data centers has become more and more serious. Therefore, energy consumption management has gradually become one of the hot issues in the field of cloud computing. For this reason, this paper aims at building a power model of servers and investigates the energy-measurement system. We propose a distributed energy consumption measurement system (abbreviated as DEM) for heterogeneous cloud environments based on a multi-component power model. We investigate the mathematical relationship between the resource usage of the key components (CPU, memory and disk) and the system energy consumption. Then we give the power modeling method of each key component. DEM can not only estimate energy consumption of heterogeneous cluster environments (Linux and Windows NT), but also support various CPU power models. In addition, we also present a disk power model that uses several thresholds to distinguish between disk sequential and random read/write status, for achieving more accurate disk power calculation. Experiments are performed on a heterogeneous cluster with workload generated by *PCMark* and *Sysbench*. The results show that the proposed DEM system can not only achieve measure the energy consumption of heterogeneous cloud servers, but also have great accuracy on estimating cluster energy consumption.

Key words: cloud server, heterogeneous cloud environment, power measurement, power model

1. Introduction

Statistically, worldwide data center power consumption has increased from 700 billion degrees to 3300 billion degrees from 2000 to 2007. By 2020 the figure is estimated to increase by another 1 trillion degrees [1]. Only 8.5 percent of data center executives expect data center capacity to remain adequate by 2015. By 2020, data centers will be at least twice in scale compared with that in 2010, reaching 78 billion US dollars [2]. In 2012, total data center power consumption in China reached 66.45 billion kwh, accounting for 1.8% of the national industrial electricity consumption. This value is equal to the total annual electricity consumption in Tianjin, while the Three Gorges power generation but 78.3 billion kilowatt hours [3]. The domestic data center capacity may also increase by 5~8 times in the next 5 years. On the one hand, data center energy usage is inefficient. According to statistics from the Ministry of Industry and Information Technology, the average Power Usage Effectiveness (PUE) of data centers in our country is between 2.2 and 3.0, but the actual value is more than this. For businesses, data center electricity bills have become a big part of the expense. For example, the revenues of China Unicom in 2012 was 40.7 billion dollars while its profits reached only 1.2 billion dollars with an electricity cost of 1.7 billion dollars. Therefore, there is great room for improvement in data center energy consumption management. To improve PUE of data center, we first need to monitor the energy consumption of the current data center.

In the previous study, existing methods of energy consumption can be summarized as follows [4]: Hardware-based direct measurement methods [5], Energy model-based methods, Virtualization Technology-based methods and simulation-based energy consumption estimation methods. The Hardware-based direct measurement method is mainly applied to traditional data centers. Energy model-based method is a mainstream method to calculate the energy consumption of cloud computing because of its high flexibility and fine granularity [27]. Currently, energy consumption measuring tools for heterogeneous cloud environments [24,25] are still uncommon, with most of the tools only focused on cluster resource utilization and network monitoring, like Ganglia [6] and Nagios [8]. The main research direction is using power model to estimate power consumption, and modeling CPU, memory and disk as three parts. The power model of each component can be characterized by a certain number of characteristic variables. Although the power model based on many characteristic variables has higher accuracy, it is difficult to directly obtain all the variables in actual operating systems. Therefore, a certain means of simplification is needed [4][11]. So when developing an actual energy monitoring system, we need to use the power model that both meets good accuracy and low complexity.

For the above-mentioned requirements of energy monitoring system, we present an energy consumption monitoring tool based on the power model for heterogeneous cloud computing environment [23]. The tool is named Distributed Energy Meter (abbreviated as DEM). The main contributions of this paper include: 1) With an investigation into the disk

- *Weiwei Lin (corresponding author), Haoyu Wang, Yufeng Zhang and Deyu Qi, are with the School of Computer Science & Engineering, South China University of Technology, Guangzhou, China*
-

power behaviors in sequential I/O and random I/O, we propose an improved I/O-mode aware disk power model with multiple variables and thresholds. 2) We build a dynamic adjustable CPU power model that enables configuring the CPU power model as a power function model or a linear model. Moreover, the key parameters in the CPU power model can be trained and adjusted based on the configuration of the machine to make the energy consumption calculation more accurate. 3) DEM supports energy consumption monitoring in heterogeneous cloud environments (both on Windows and Linux servers). 4) We use an adaptive resource monitoring method that combines periodic push mode with event-driven push mode in the heterogeneous computing environment.

Energy reduction have become the focus of researchers cloud computing and data center managers concerned: someone use energy consumption monitoring to overall optimize the management and deployment of cloud computing applications to reduce carbon dioxide emissions [33]; Adhinarayanan et al. [34] analyzes energy consumption for specific applications, such as visualization processes, to optimize the calculation process and deployment. For similar energy management issues in heterogeneous cloud computing environment, DEM can provide an effective solution. The problem of energy consumption and its optimization technology is very important. In addition to energy consumption in cloud computing, there are similar problems in related fields [28-32]. We can learn from their methods from these studies [28, 33].

The paper is organized as follows. The second section of this article mainly introduces the server power model and relevant research on cluster energy monitor system. The third section introduces the system power model adopted by DEM, and the fourth section gives the key technologies in the design and implementation of cloud server energy consumption measurement system, including system architecture, communication design, data design, implementation of Master and Slave nodes. The fifth section gives the related system verification and analysis and we conclude our study in section 6.

2. Related Work

At present, most existing cluster monitoring systems focus on resource utilization monitoring, especially for the performance monitoring of homogeneous clusters such as Hadoop and Spark. The challenge of energy consumption measurement system in heterogeneous cloud environment is the rationality of system architecture design and the accuracy of power model. We will introduce the related work from power model and energy consumption measurement system two part.

2.1 Power Model

For both stand-alone and cluster energy consumption measurement tools, the most important thing is the accuracy and adaptability of the built-in power model. The higher accuracy makes the result of software measurement more valuable, and the stronger adaptability makes the model match the more hardware models with the lowest possible complexity.

Basmadjian et al [18] pointed out that for servers with local storage, CPU consumes about 37% of the energy, while memory, motherboard and disk consume 17%, 12% and 6% respectively. Therefore, we mainly address on the CPU, memory and disk these three major components. For CPU energy models, linear and non-linear estimation methods are commonly used. In previous study [18], linear model is used to estimate the energy consumption of the CPU. Whereas, Hsu et al [17] pointed out that the linear model has a better result for the early absence of Hyper-Threading and Turbo Boost technology modes. However, as CPU manufacturer technology advances, the error of the linear model becomes larger. Also in that paper, the author calculated the error of energy consumption estimation including the linear model, the polynomial model and the power function model by using the statistical and regression methods on more than 100 data of the SPEC website. Results show that the power model has achieved good accuracy. In the study of CPU energy consumption performance under VM environment, there is also a gap between the model that expresses the power function model and the linear model [26].

According to the formula proposed by Janzen et al [19], the power consumption of memory is closely related to the running state, operating voltage and many other constant parameters. Although the power consumption calculation model proposed in the literature is very accurate, it is difficult to be practical due to the challenge of obtaining those parameters form the current operating system. Literature et al [11] proposed a more concise energy consumption calculation formula, using the last layer cache miss rate (LLCM) to characterize the memory activity and thus to estimate the memory consumption. However, for this method, the value of the LLCM counter is equally difficult to obtain in the system.

The working status of the disk device [20] can be described by the disk rotation rate, the average query distance, the average query time, etc. So the power model of disk can also described by similar data [21]: revolutions per minute (RPM), disk radius and buffer size, etc. Basmadjian also discuss the power model that based on read-write probability and idle probability[18]. However, there are still many challenges to obtain all variables of that model in practical use.

2.2 Energy Measurement Tools

2.2.1 CloudMonitor

CloudMonitor is an energy monitoring tool based on the energy model [9], it advises the deployment of the cluster and requires no additional hardware support. The software uses a methodology based on work done by Bohra and

Chaundary in the paper VMeter [10], which predicts energy consumption by monitoring the amount of hardware resources used on the computer, including the CPU, cache, RAM, Disk, and driver. The proposed power model is based on the linear relationship of the system subcomponents. The power model described as follows:

$$P_{\{CPU,\ cache\}} = a_1 + a_2 p_{CPU} + a_3 P_{cache} \tag{1}$$

$$P_{\{DRAM,\ disk\}} = a_4 + a_5 p_{DRAM} + a_6 P_{disk} \tag{2}$$

$$P_{total} = \alpha P_{\{CPU,\ cache\}} + \beta P_{\{DRAM,\ disk\}} \tag{3}$$

$\alpha_1$and α_4 are system idle power. α_2, α_3, α_5 and α_6 are weights. $\mathrm{P_{CPU}}$, P_{cache} , $\mathrm{P_{DRAM}}$ and $\mathrm{P_{disk}}$ are system events that produced by CPU. $P_{\{CPU,\ cache\}}$ and $P_{\{DRAM,\ disk\}}$represent {CPU, cache} and {disk, DRAM} two subsystems power. All weights including α and β are manually configured according to different workloads.

2.2.2 Joulemeter

Joulemeter [11] is a tool with multiple power models for measuring the power consumption of virtual machines, servers, desktops, laptops and individual processes. It provides visualized power distribution data that provides useful guidance for data center power budget settings for virtual machines and battery management for mobile phones.

Joulemeter decomposes system into various components to build the system power consumption model. Developers of Joulemeter collected a large amount of measured data for model learning. Joulemeter obtains the corresponding parameter information by measuring hardware resources (CPU, disk, memory, screen, etc.) usage. And then it takes those values into the corresponding term of the power consumption formula to estimate the current system power consumption.

3. System Energy Model

The build-in power consumption formula of EM basically includes three component power models: CPU model, memory model and disk model. At the present stage, the basic idea is to model the energy consumption of important components. The formula of overall energy consumption is：

$$E_{total} = E_{fix} + E_{storage} + E_{comp} \tag{4}$$

$E_{storage}$ represents the energy consumption of disk and memory， E_{comp} represents energy consumption of CPU. The energy consumption of other components (e.g., network communication energy consumption) in the cluster is mainly produced by network interface cards (NICs) and other external independent network switching equipment such as routers, switches. We do not separately estimate the power of networking activities as the power consumption of the external independent network switching equipment is difficult to obtain by software. Besides, the absolute value and fluctuation of NIC power consumption is small -- It was observed that the fluctuation of 1Gbps Ethernet card is negligible in the test [17]. Likewise, for other components, so we choose not to design their power consumption model specifically. The energy consumption of these components is included in E_{fix} as a static value.

3.1 CPU Model

The power of CPU is closely related to CPU performance status (P-states). The P-state is determined by activity status, execution of specific instructions, cache usage, and frequency thresholds. Using the above variables during a CPU runtime to model CPU power consumption can achieve very high accuracy. However, the above theoretical modeling requires a complete understanding of the CPU hardware architecture and a large computational overhead. Many researchers choose to determine P-state by tracking CPU operation and sleep time. The operation and sleep time ratio can be presented by CPU utilization. So in the early studies [12][17], the CPU power consumption for a given frequency was generally calculated using the linear model shown in Equation (5):

$$P_{cpuLinearModel} = P_{idle}^{cpu} + \left(P_{peak}^{cpu} - P_{idle}^{cpu}\right)U \tag{5}$$

$\mathrm{P}_{\mathrm{idle}}^{cpu}$ represent CPU idle power， $\mathrm{P}_{\mathrm{peak}}^{cpu}$ represent peak power and U is utilization of CPU. For recent years, CPU manufacturers add new technology like Hyper-threading and TurboBoost to CPU, making the recent CPU model has dynamic frequency and other new features. Simple linear model cannot suit to those CPU model. We do a power test for a server CPU model that has four physical cores with linear model:

To make DEM easier to deploy and without more power experimentation under different CPU utilization prior, we choose to model only through the $\mathrm{P}_{\mathrm{idle}}^{cpu}$ and $\mathrm{P}_{\mathrm{peak}}^{cpu}$ two endpoints. As Figure 1 shows, if only using two points, the linear model result will be lower than actual value.

In Hsu et al and our previous study [12][17], polynomial model can be another choice. However, there are three shortcomings in practical application: 1) A polynomial model with at least three parameters cannot be fitted by only two endpoints. 2) Even if more experiments can be performed to obtain more intermediate values, multiple iterations are needed in the fitting process, which is likely to fall into the local optimal solution. 3) The polynomial model fitting results are also prone to over-fitting.

The above literatures also mention the power function model can meet the accuracy requirements and with less complexity [12]:

$$\mathrm{P}_{cpuPowerModel} = P_{idle}^{cpu} + \left(P_{peak}^{cpu} - P_{idle}^{cpu}\right)U^{\beta} \tag{6}$$

where β is an exponential of the power function model. Using the two end-point values, we obtain the fitted β value and compare it with a linear model using a power function model:

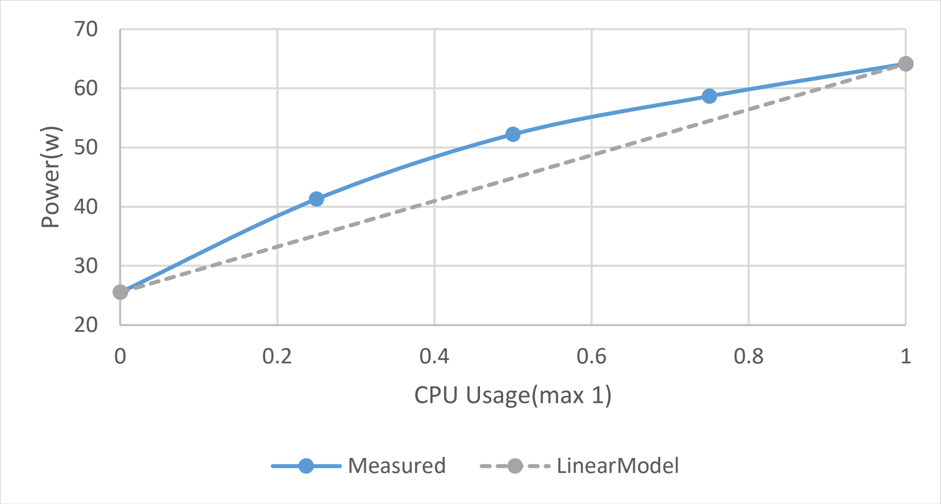

Figure 1. CPU linear model

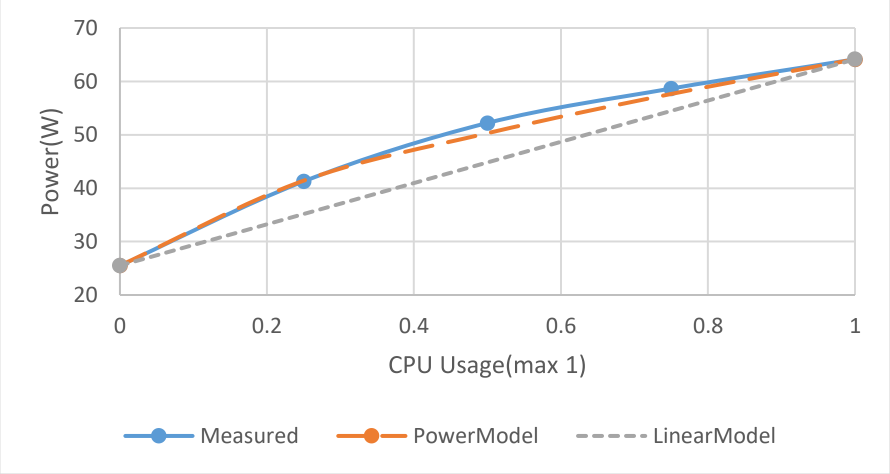

Figure 2.CPU power model

Figure 2 shows that the fitted power function model has higher accuracy. As for the value of β, Hsu et al [17] pointed out that the exponential values of the power function models corresponding to CPU models at different periods are different. After counting the power consumption curves of 177 CPU models on the SPEC POWER website, and they point out that exponential value from 1 to start declining. Therefore, we use the β-adjustable power function model to model CPU. When the server is using an older CPU model, the value of β can be set to 1, which point the power function model of Equation (6) degenerates into a linear model. When there are many servers in the cluster with the same CPU type, deployment staff can perform energy consumption tests on one of them and adjust the values of $\mathrm{P}_{\mathrm{peak}}^{cpu}$ and β so that the DEM can have well result.

3.2 Memory Model

Memory energy consumption is mainly generated by the operations of memory read/write and page swapping. Theoretically, the swap rate or the last level cache misses (LLCM) can describe memory activities more accurately [11]:

$$P_{mem} = P_{idle}^{mem} + C \cdot N_{LLCM} \tag{7}$$

N_{LLCM} represent LLCM, C is the constant to be trained. These metrics are difficult to obtain in a Host-OS and virtual machine environment. As an alternative, DEM uses the current amount of available memory to measure the current memory load. It is based on an idea that higher memory usage means more frequent page swapping in / out. The memory model is designed as the following:

$$P_{mem} = P_{idle}^{mem} + C_m \cdot U_{mem} \tag{8}$$

U_{mem} is the current system memory footprint, and its unit is GB. C_m is a fixed constant associated with memory configuration and can be obtained by training.

3.3 Disk Model

For frequently-executed I/O-intensive workload servers, disk energy consumption accounts for a large percentage of the total system energy consumption, so the accuracy of the disk power model is important. Disk energy consumption is mainly due to magnetic head read, write and rotation. Bostoen et al [35] proposed to consider two key disk operations: query and data transfer, they proposed a disk-dependent linear disk power model:

$$\mathrm{E}_{disk}(T) = P_{disk\_idle}T_{idle} + (P_{disk\_max} - P_{disk\_idle})(T_{sk} + T_{tf}) \tag{9}$$

T_{idle},T_{sk} and T_{tf} represent the disk idle, query and transmission time. Although the model distinguishes the disk query and transfer operations, but did not reveal the difference between the two in generating energy consumption. Current operation system is also difficult to provide counters access to the corresponding query and transmission time.

Kansal et al [11] proposed that using read and write bytes to estimate disk energy consumption. This model is essentially linear.

$$E_{Disk,A} = \alpha_{rb} * b_{r,A} + \alpha_{w,b}b_{w,A} \tag{10}$$

$b_{r,A}$ and $b_{w,A}$ on behalf of the disk read and write the number of bytes. At the same time, through further experiments, they found that the difference in energy consumption of reading and writing unit bytes is very small. So b_{io} is used to represent the sum of the number of read and write bytes, making equation 10 more simpler:

$$E_{Disk}(T) = \alpha_{io}b_{io} + C_{Disk} \tag{11}$$

We argue that the power consumption of disk is not only related to read or write bytes, but also associated with I/O mode. We run the experiment on an ordinary desktop machine disk Seagate ST31000340NS 1TB SCSI 7200RPM SATA-II. *IOmeter* [16] was used to carry out the correlation parameter test. The transmission test block size in the test was set to 64KB. When the transport block size is set too small, it is likely to consume a large fraction of processor time, which consequently affects the system performance and increases the additional CPU power consumption. When the transferred data block is greater than 64KB, the operating system I/O subsystem will divide it into multiple 64KB data blocks. So considering the actual application scenarios we in the experiment set the *I/Osize* to 64KB. There are two modes for disk read and write: sequential read-write and random read-write. In this paper, both are investigated together with disk power, s (I/O speed) and o (I/O operations). The abscissa values in Fig. 3, Fig. 4 and Fig. 5 are read and write ratios.

By looking into Fig. 3, Fig. 4 Fig. 5, we can find the following key features of disk.

Sequential I/O mode:

- Disk I/O speed: in pure read and pure write, disk I/O speed is significant larger than that in mixed read-write situation. And in mixed read-write situation, I/O speed has little difference.
- Disk I/O operations: the situation is similar to I/O speed.
- Disk power consumption: significant power consumption at both ends, proportional to I/O throughout.
- Random I/O mode:
- Disk I/O speed: at different read-write ratios, the I/O speed does not differ much.
- Disk I/O operations: similar to I/O speed, the value of I/O operations changes little in tests with different mixed read-write ratio.
- Disk power consumption: roughly, it is proportional to I/O speed.

We propose two thresholds to distinguish disk sequential I/O and random I/O modes using I/O operations per second and read/write speed: *Hs* and *Ho*.

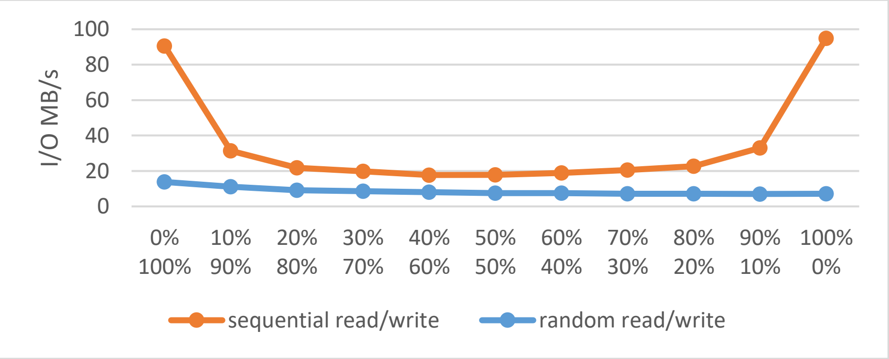

Figure 3. Disk I/O speed in sequential read-write and random read-write

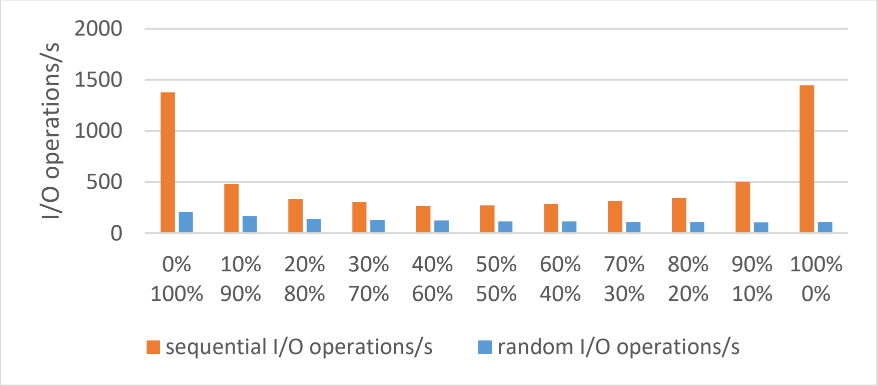

Figure 4. Disk I/O operations in sequential read-write and random read-write

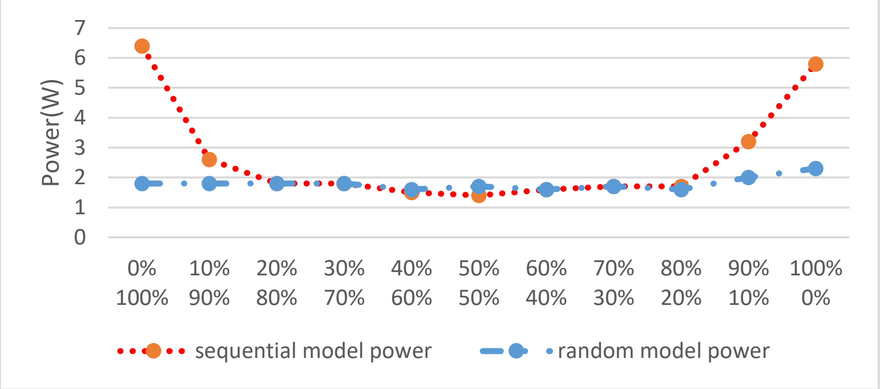

Figure 5. Disk power in sequential read-write and random read-write

Although the power consumption of a disk is very high in sequential read and write, but considering the actual production load, the disk will rarely be in such a single working state. So we also consider the case of mixed read and write. The data demonstrated in Fig. 6 is obtained from the disk power in Fig. 5 divided by the I/O speed in Fig. 3. The line chart in Fig. 6 shows noticeable difference of α value in two modes. α_{seq} and $\alpha_{\mathrm{rnd}}$are average value units I/O speed's power in each mode.

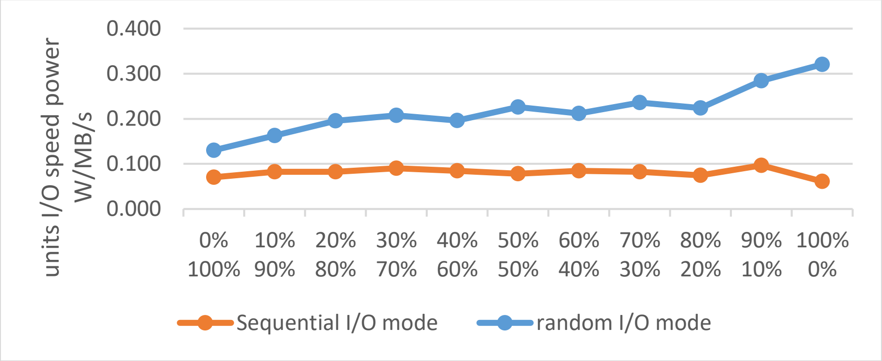

Figure 6.Disk power of unit I/O speed

Form the above results we can indicate that it is quite necessary to distinguish between different I/O modes: sequential I/O and random I/O. Therefore, we propose a power model based on multivariate thresholds and distinguishing I/O modes for mechanical disk (HDD) as follow:

$$P_{disk} = \begin{cases} \alpha_{seq} \cdot s, if\ s > H_s \text{and } o > H_o \\ \alpha_{rnd} \cdot s, \text{otherwise} \end{cases} \tag{12}$$

$$s = s_{read} + s_{write} \tag{13}$$

$$o = o_{read} + o_{write} \tag{14}$$

α_{seq} and α_{rnd} represent parameters corresponding to the two disk I/O modes: sequential I/O and random I/O. *d* denotes I/O speed (MB/s) and o denotes disk operations per second. In these two different modes, the tested disk showed a very large gap in performance regarding I/O speed and operations per second. For instance, in random I/O mode, even if the number of I/O operations per second is significantly lower than that in sequential read-write mode, the according disk power consumptions show little difference. We also observed that energy consumption of disk shows little difference in 100% read and 100% write in the same I/O mode. So there is no need to distinguish read and write for s and o. *Hs* and *Ho* are two thresholds parameters used for determining the current disk I/O mode. For a single disk or disk array (such as RAID0 / RAID1, etc.), data center managers can adjust the values of H_s and $H_o$according to the system storage performance to make the disk energy consumption estimation more accurate.

4. DEM Design and Implementation

4.1 DEM Design

DEM use a typical Master-Slave architecture, as shown in Fig. 6. Master statistics and displays the energy consumption of the entire cluster, and checks CPU, memory and disk running state of every single slave. Each Slave runs on Windows or Linux operating system and measures and monitors the energy consumption of deployed cloud servers. At the same time Slaves monitor the CPU, memory, Disk usage, and applies the default utilization warning threshold to reporting whether the Master alarm is triggered. The communication between Master and Slave is based on TCP socket

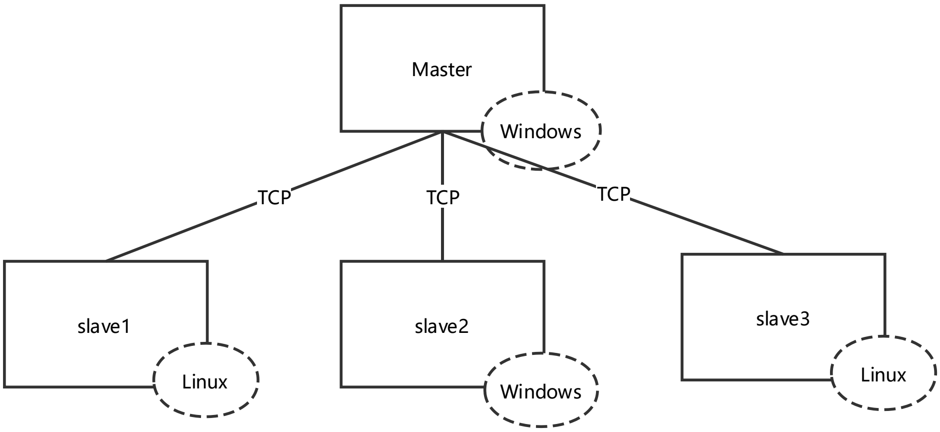

Figure 7.DEM architecture

4.1.1 Slave Design

As shown in Figure 7, the slave node mainly comprises six modules: hardware detection module, model matching module, resource monitoring module, power estimation module, data sending module and data persistence module.

The hardware detection module detects the hardware of the local device to obtain the hardware model. The model matching module matches the same or the closest hardware model with energy consumption information according to the underlying hardware. The power estimation module uses the method based on component power models to calculate the energy consumption. The measured energy consumption information is sent to the data sending module and the persistence module for network communication and persistence.

4.1.2 Master Design

As shown in Figure 8, the master node of the DEM is mainly composed of 6 modules: network communication module (including two submodules of a periodic data receiving module and an event message interaction module), cluster maintenance module, data statistics module, data display module, persistence module and query module.

The network communication module, as the core function module of the Master, is mainly responsible for handling the periodic data (energy consumption information and resource utilization) and event interaction transmitted by the slaves. Based on the information obtained by the event interaction module, the cluster maintenance module maintains a list of connected cluster nodes, a list of downed nodes and a list of cluster nodes that are overloaded with alarms. The data statistics module collects energy consumption information from the cluster to obtain information such as maximum power consumption and maximum CPU utilization. The statistical data will be put into RRD (Round Robin Database) database by the persistence module, meanwhile it records log information. The data display module displays the statistical energy consumption information or the resource utilization rate and obtains the historical energy consumption information of the cluster to the query module.

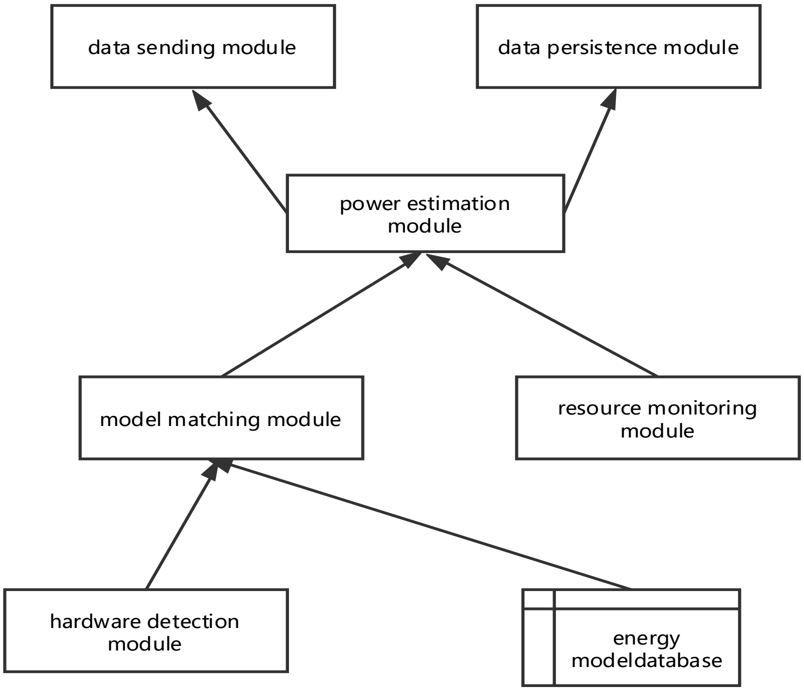

Figure 8.DEM-Slave architecture

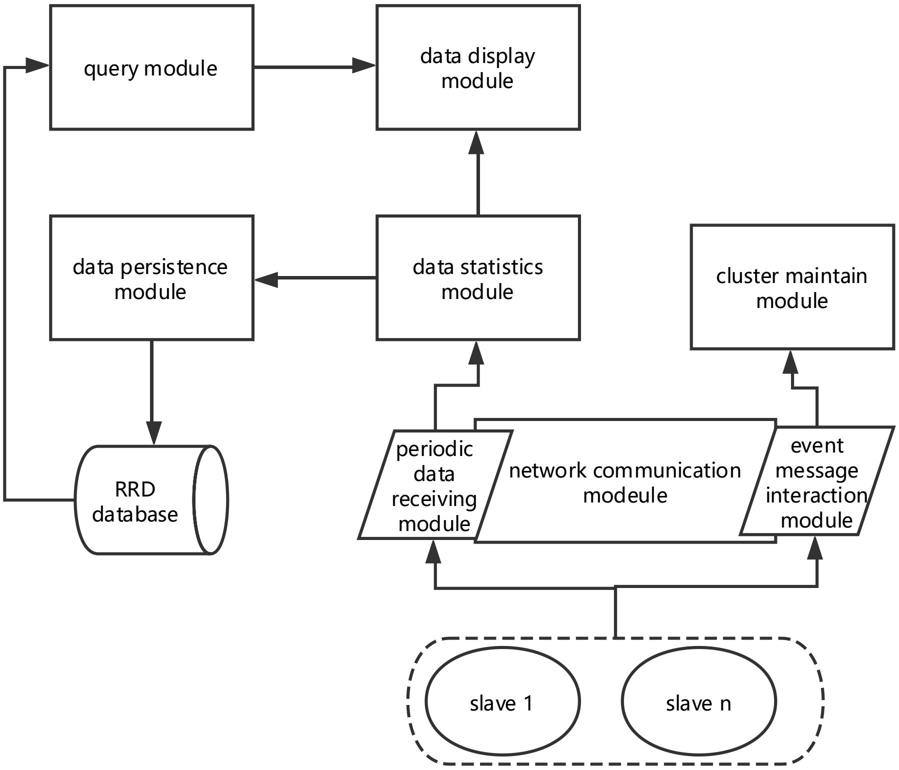

Figure 9.DEM-Master architecture

4.1.3 Communication Design

In DEM, we use two kinds of data packets: Static information data packet and dynamic information data packet: Static information data packets belong to the following categories:

- Slave Request Packet: Contains the operating system type, CPU model, memory model and size, disk model and size, startup time, and IP address of the current Slave node.
- Master request reply packet: contains whether to allow the slave nodes to join using a boolean response information.
- Master RRD Data Request Package: contains requests to specify the RRD history database for the Slave node.
- Slave RRD response packet: contains RRD database information of the Slave node.
- Slave alarm packet: contains specific alarm information (such as CPU load is too high) and IP address.

Dynamic information data packets have only one class:

- Real-time data packet: contains the energy consumption information (including detailed energy consumption information of each component) from the slave nodes at the corresponding time interval, resource utilization information of each component, and transmission time interval.

At the same time, we propose a self-adaptive heterogeneous cluster energy consumption information monitoring method. Compared with the software or system mentioned in Section 2, this method combines the periodic push and the event-triggered push to obtain the slave node's data. The periodic push mode shown in Fig. 10 refers to that the slave node pushes the dynamic information data packet to the master node at a set interval. Fig. 11 shows the event-triggered push mode where the slave nodes send the corresponding static data packet to the master node after receiving a specific request or meeting certain trigger conditions (such as a high load alarm).

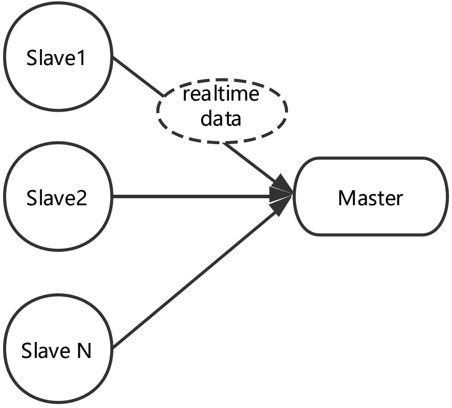

Figure 10. periodic push

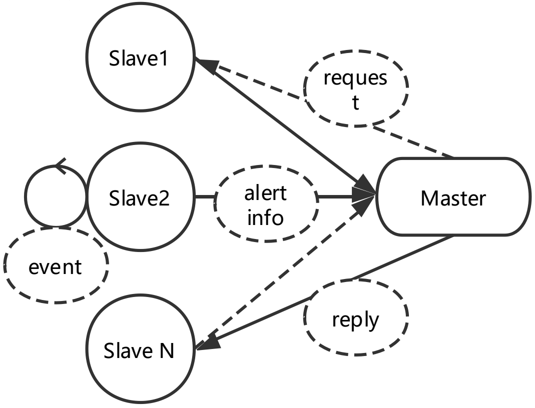

Figure 11.event-triggered push

By combining these two modes of communication, we can get a list of the methods of distributed monitoring. The third column in Table 1 shows the consistency between the master statistics and the actual data collected by the slave nodes. Adaptability indicates how well the method responds to the latest changes in the slave nodes. Overhead reveals the size of data to be transmitted during the monitoring process the number of packages.

Table 1.Monitor method

Method	Explanation	Consistency	Adaptability	Overhead
P-Push	periodic push	Good	Bad	n-1
E-Push	event-triggered push	Very good	Good	n

4.2 DEM Implementation

With the goal of cross-platform design, both Master and Slave in DEM systems are based on Qt implementations. The TCP / IP communication used by the network communication part is also implemented by qTcpSocket in the Qt library. This section mainly introduces the implementation of Slave, Master and communication.

4.2.1 Slave Implementation

- Hardware Detection Module: The hardware detection module of Slave has different implementations on different platforms. WMI (Windows Management Instrumentation) service is used on Windows, and on Linux, the raw data of the file about hardware information in the /proc virtual directory is read to obtain the hardware information.

In the Windows version, we obtain static component information about systems, applications, and hardware devices provided by WMI based on the CIM standard, such as system type, CPU model, memory capacity and model number, disk capacity and model number. In the Linux version, slave reads */proc/stat, /proc/meminfo* and */sys/block/sda/stat* files to obtain CPU, memory and disk usage information.

- Model Matching Module: When the DEM cannot determine the P_{peak}^{cpu} value of the CPU model, it needs to approximate the peak energy consumption of the CPU using the thermal design power (TDP). Slave builds and maintains a comprehensive database of system component models. Component information and corresponding parameters are stored as database records. DEM uses the extensible markup language to construct the component database in order to promote the efficiency of string matching. DEM maintains two model databases: CPU database and disk database. Memory typically consumes not more than 28% of system total energy. Besides, different memory brands and models make little difference in energy consumption characteristics. Thus, we directly use the memory utilization to estimate memory energy consumption. EM records all models in ModelList.xml. The CPU model entries are stored within the <*CpuModelList*> tag. The sub-tags are the manufacturer (such as Intel or AMD) and the processor family (such as the Desktop or Server series). The Disk Model entry is stored within the <*DiskModelList*> tag. Each disk model is quoted in separate <*Record*> tags

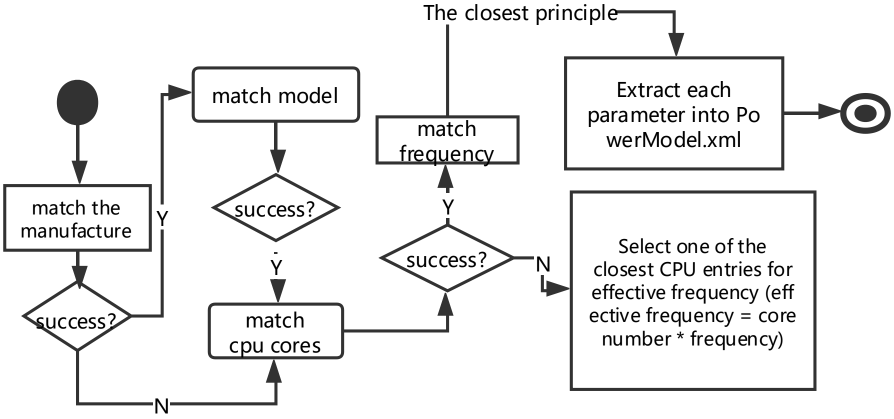
The closest principle
match model
match frequency
Extract each parameter into PowerModel.xml
match the manufacture
Y
success?
Y
Y
success?
N
Select one of the closest CPU entries for effective frequency (effective frequency = core number * frequency)
success?
match cpu cores
N

Figure 12. the match process of CPU entries

Slave detects whether the current environment is running the first time. If the last-time hardware test results are saved, then slave directly reads all the parameters recorded in the PowerModel.xml. If slave is running at the first time in the current environment, it turns to read ModelList.xml and match the hardware entries. If an entry in the database matches the current hardware model, the relevant model parameters will be extracted; otherwise, the parameters need to be estimated. Estimation algorithm is simple: Find the most similar entries according to performance related metrics (such as CPU frequency). Take CPU matching process as an example, the detailed algorithm is shown in Figure 12.

- Resource Monitoring Module: Resource utilization monitoring on Windows also uses the performance counters (PDH) in the NT kernel. Monitoring on Linux relies on reading the file information in the / proc file system to obtain the change information.

Performance monitoring is implemented via a Windows NT build-in system tool. Windows NT has always been integrated with performance monitoring tool that provides information on the current operating system status. For a variety of object, it provides with hundreds of performance counters. Windows performance counters can be called from PDH function. Each performance counter has its own detection performance object including Processor, Process, Memory, Physical Disk, etc. Counters typically stored performance-related information about operating systems, applications, services and drivers. They are used to analyze system bottlenecks and optimize system or application performance.

Linux monitoring needs to read the raw data of the kernel file information and calculates the difference between the raw data of the current moment and of the previous moment. Then it calculates the formula based on the utilization of different components to get the utilization value.

4.2.2 Master Implementation

- Network Communication Module: The network communication module consists of two sub-modules. The periodic data receiving module is responsible for processing data when the real-time energy consumption data of the cluster is collected. The event message interaction module processes the information triggered by events including adding, leaving and alarming of the slave. Considering the cluster network environment is relatively stable and the communication process will be transmitting RRD database within a larger content file, so we choose more stable and reliable TCP communication instead of UDP communication.

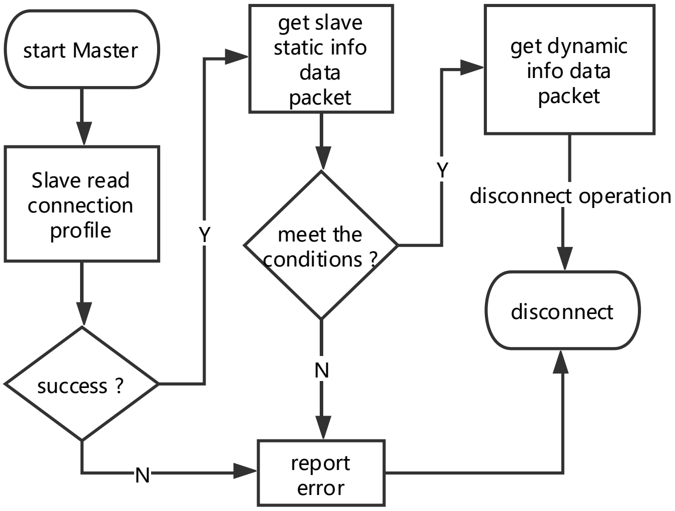
start Master
Slave read connection profile
success ?
Y
N
get slave static info data packet
meet the conditions ?
Y
N
report error
get dynamic info data packet
disconnect operation
disconnect

Figure 13. Master connection process

The connection process shown in Fig. 13, Master first starts listening a port, Slave reads the configured network connection file to connect. If connection is successful, the Slave will first send a request with its own hardware information static packet. After the Master receives the request message, it determines whether to allow the Slave to join according to the setting criteria. The judgment information includes whether the number of connected Slave nodes has reached the maximum, whether the IP address of the Slave node is in the black list, or other information that does not meet the requirements of the software and hardware. After meeting the conditions to join the monitoring cluster, the slave periodically transmits the measured local energy consumption information to the master node. If receiving a Master request or triggering a monitoring alarm, the Slave node will also transmit the corresponding data to the Master node.

– Data Statistics Module: The function of data statistics module is to calculate the dynamic information data packets transmitted by nodes. The calculated data includes the average system power consumption, average CPU power consumption, memory power consumption, and average disk speed of the current cluster. At the same time, the node with the highest power consumption in the current cluster is counted, and the node with the largest CPU, memory, and disk utilization is counted.
– Persistence module: The Master uses the RRD (Round Robin Database) database, which operates using the RRDTool. RRD database has a fixed size that can be set, the data can be compressed by the aggregation operation, and it is suitable for monitoring the system data acquisition or log storage. At the same time, RRDTool is also a powerful drawing engine, and many tools such as MRTG can call RRDTool to plot. The various variables calculated in the data statistics module use the RRDTool to persist in the master node. The Master initiator can set a custom data aggregation parameter before the RRD database is initialized
– Data display module: In addition to displaying the statistics data in the data statistics module, the query module can also obtain the historical information stored in the RRD. Using RRDTool to graph historical changes, cluster managers can collect the energy consumption performance of a cluster under a specific load more directly.

5. System Validation

This section presents our experiments to evaluate DEM, including CPU power estimation experiment, disk power estimation experiment and cluster power estimation experiments.

5.1 Experiments Setup

Table 2. CPU experiment parameter

Parameter	Value
C_m	0.3 W/GB
α_{seq}	0.07 W/MB/s
α_{rnd}	0.22 W/MB/s
H_o	150 operation/s
H_s	15.0 MB/s

The machine used in CPU and disk power experiments is the Dell T110 II Tower Server, which is configured for Intel Xeon E3-1220 V2 @ 3.10Ghz, 8GB RAM, Seagate ST2000DM006 2TB 7200RPM SATA-III. The load generation software is a related test suite in the ***PCMark 7 professional Edition v1.4.0***. The external electric meter used in the test is ***Watts Up?pro***, which can store and record the related data of energy.

The parameters in Table 2 are obtained from the tower servers. The same experimental parameters are used in subsequent distributed experiments due to the universality of general DDR3 memory and SATA mechanical hard disks.

Table 3. Machine config in cluster power estimation experiment

Machine model	CPU	Operating system	$\mathrm{P}_{\mathrm{peak}}^{cpu}$
ThinkPad X230	I5 3320M	Ubuntu 17.04	21.47
Lenovo Y50	I5 4200H	Ubuntu 16.04	28.9
Dell T110	E3 1220 V2	Windows server 2008 R2	38.63
Dell R730	E5 2603 V3 *2	Windows server 2008 R2	17.71

Table 3 shows the machines used in the power estimation experiment of the small cluster. There are 4 Slave nodes in total, including PCs, tower servers, and blade servers three different machine types. The experimental environment includes a heterogeneous system environment composed of Windows and Linux. As for the load-generate software, productivity suit is used on Windows systems with ***PCMark 7*** and ***sysbench*** is used on Linux systems.

5.2 CPU Power Model Experiment

The benchmarking suite for CPU test is Computation Suite including three sub-tests: Video transcoding-downscaling, video transcoding-high quality and Image manipulation. By recording the CPU power consumption in running these three CPU-intensive applications, the accuracy of CPU power consumption model can be effectively detected.

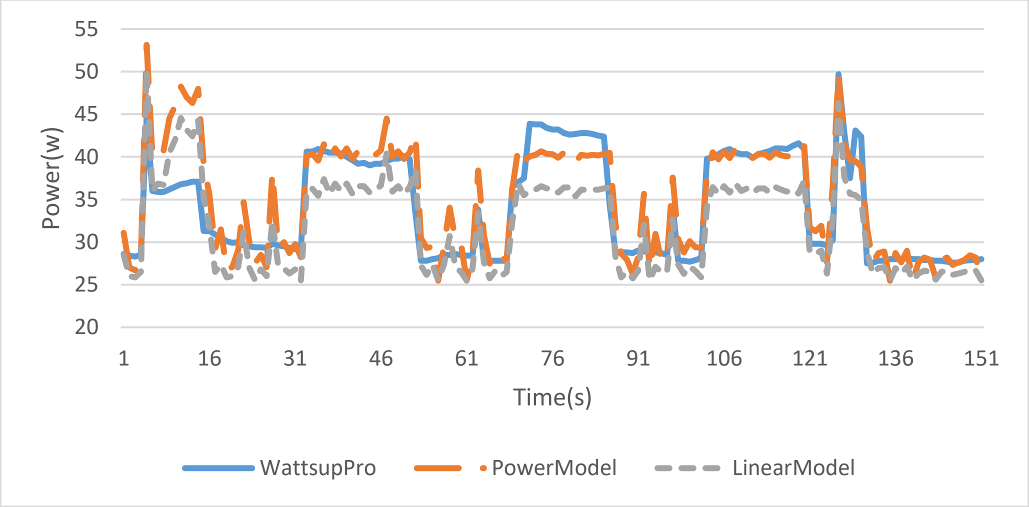

Figure 14. CPU power estimation of linear-power model

Again, we compare the linear and power function models mentioned in Section 3.1. From Figure 14, we can see that the estimated value of power function model is higher than that of linear model. The reason is that the CPU power estimation relies on two fixed values P_{idle}^{cpu} and P_{peak}^{cpu}, which is a straight line connecting two points for the linear model. However, the power function model has a "bump", while the actual CPU power consumption will have a "bulge" process as the utilization rate increasing [15]. As shown in Figure 1 in Section 3, the linear model is completely below the power function model and the actual value in the intermediate stage. Although more data can be used to further fit the linear model to make the estimation error lower, this increases the complexity of using the DEM system. So the power function model simplifies the complexity of the model while ensuring the accuracy rate. Under the complex CPU load test set shown in Figure 14, the average positive-negative relative error of the linear model is -6.24%, and the average relative error is 8.89%. The average positive and negative relative power error of power function model is 2.46%, and the average relative error is 6.46%.

As we mentioned in Section 3.1, the power function is adjustable in value, and Hsu et al[17] found that the exponential value in the power function model is as close to 0.6. Thus, we started with 0.6 and experimented with 0.05 intervals to find the best fit.

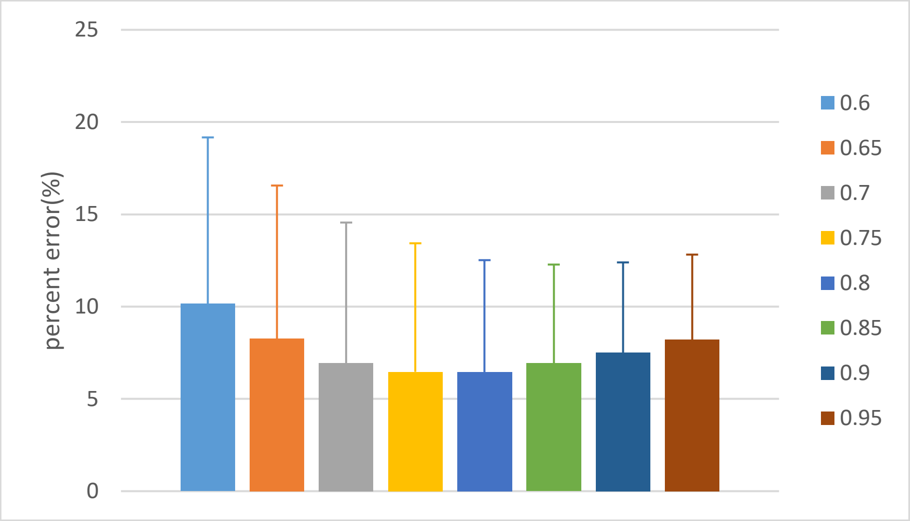

Figure 15. Exponential value experiment(Error bars shows standard deviation)

As shown in Figure 15, the lowest average relative error is close to 6.46% when the exponential value is 0.75 or 0.8. For the tower server, the standard deviation of the power function model is lower while the exponential value is 0.8. But Hsu et al pointed out that the value of 0.75 is more in line with the trend of change and is more universal. So in the following experiments, we set the value of β to 0.75.

5.3 Disk Power Model Experiment

The disk experiment suite is named System Storage Suite including Windows Defender, implementing pictures. Because a part of the disk test imposes workload on CPU, a variation in power consumption can be seen Figure 16. At the same time, we investigate the errors in power estimation for our disk model as well as the refined model that proposed by Joulemeter (Formula 11):

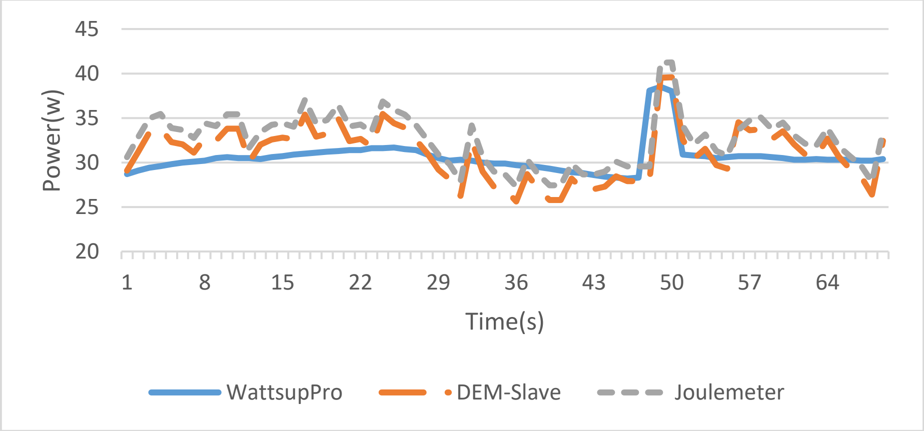

Figure 16. Disk power model experiment

As shown in Figure 16, the disk model presented in this paper is more accurate than the linear model used in most literatures. The average relative error of the Joulemeter model is 8.76%, while the average relative error of the disk model presented in this paper is 6.7%. The reason why the error is high in the early stage is that CPU produced more energy during the Windows Defender scanning.

5.4 Cluster Experiment

In the cluster experiment, the load-generating test suite used on Windows system is productivity suit, including four groups of sub-test items: text editing, web browsing and decrypting, System Storage - Windows defender and System storage-start applications. Linux systems use CPU performance testing, disk I/O testing, and scheduling performance testing in *Sysbench*. In this experiment, we also make a comparison with the CPU linear model.

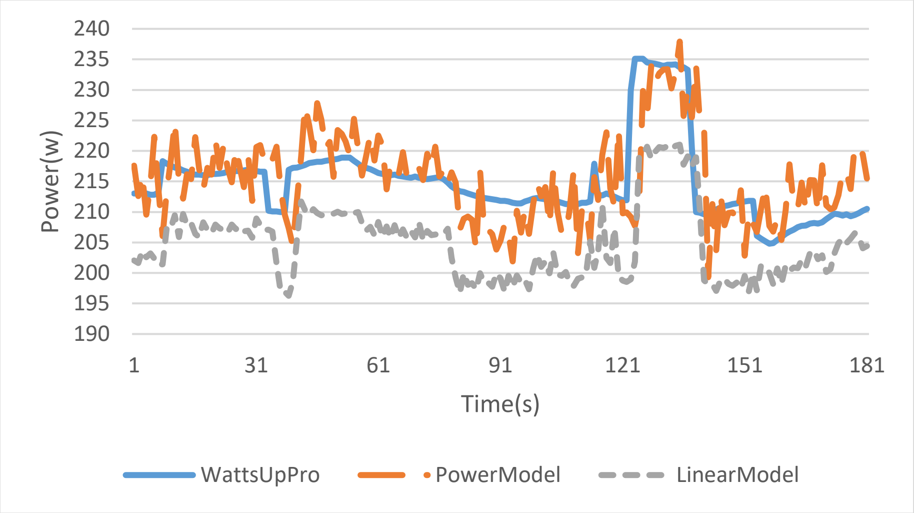

Figure 17. Cluster experiment of DEM

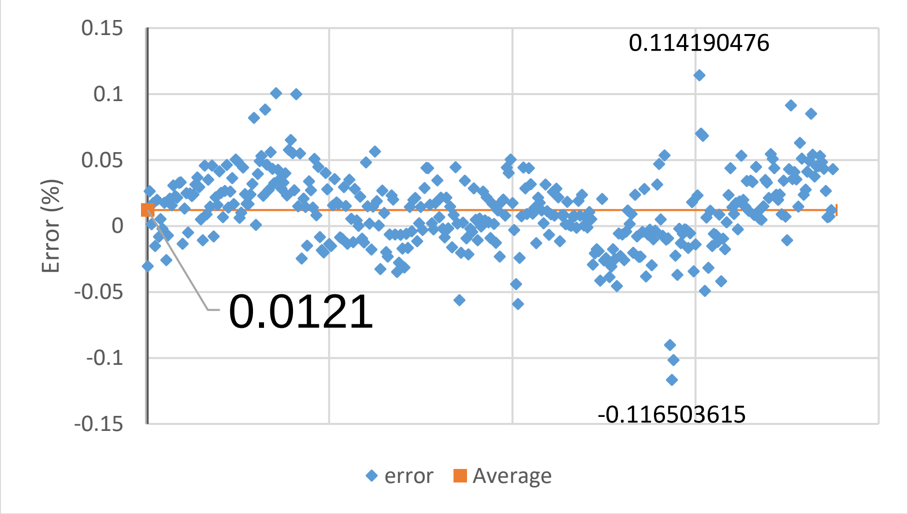

Figure 18. Standard error of power function model

Based on the standard error graphs of different CPU computing models in Figure 18 and Figure 19, The average positive-negative error of power estimation of DEM system is + 1.21%, the average relative error is 2.39% and the maximum absolute error is 11.65% when the power function model is used. When using linear model, the average positive-negative error of power estimation of DEM system is -3,93%, the average relative error is 4.02%, the maximum absolute error is 14.14%.

At the same time, we can see from Figure 17 that due to the network transmission delay and master calculation time-consuming, the energy data recorded by master node has a short lag compared with actual data. When the cluster load suddenly changed dramatically, it will cause a large error in a short time. When the cluster is in a relatively stable load state, the error of power estimation of the cluster by using the power function model is mostly within 5%.

The hardware environment of cluster power estimation experiment includes PC, Tower server and blade server. Slave nodes contain heterogeneous operating system, and load generation project is a comprehensive. When the experimental environment complexity is higher than that of *CloudMonitor*, the average relative error of energy consumption estimated by DEM with power function model is 2.39%, which is better than 4.31% of *CloudMonitor*. At the same time, the impact of DEM-slave on the monitoring node itself is very small, CPU utilization is less than 5% in the process of use.

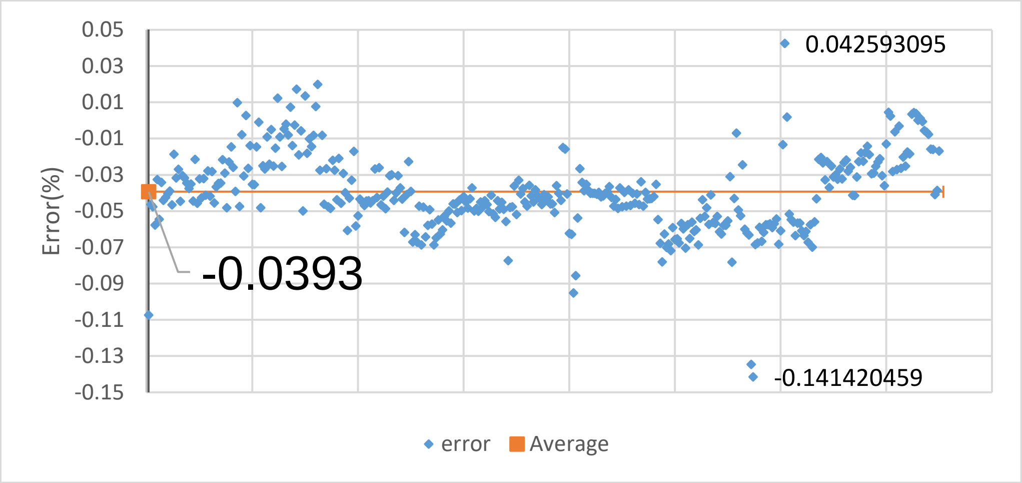

Figure 19. standard error of linear model

6. Conclusion

In this paper, we addressed the challenges in energy consumption measurement methods and the limitations of related systems. Then an energy estimation method based on multi-component energy consumption model is proposed and an implementation (DEM) is introduced. For the slave node, apart from adopting exponent-adjustable CPU power function, we propose an improved I/O-mode aware disk power model considering the difference in disk power behaviors between sequential I/O and random I/O. For the Master node, after analyzing and comparing periodic-push mode with event-triggered push mode, we choose to use a hybrid communication method that combines both periodic push and event-triggered push. The results show that the average relative error is only 2.39% under the mixed workload in heterogeneous cloud computing environment. The estimation accuracy is better than *CloudMonitor* and most other stand-alone power monitoring software.

DEM enables users to well manage the cluster by measuring and monitoring energy consumption of the cluster. DEM not only has higher accuracy in real-time cluster power estimation, but also leverages RRD database to collect and manage historical data. DEM also supports heterogeneous cloud environment with highly scalable deployment. The current version DEM still has some limitations and weaknesses, such as data security challenges. There are some recent outstanding research results on Cloud Security [38, 39] and we will try to apply these results and deep learning techniques [36, 37] to our DEM in the future. We will also extend the display module of the Master and adjust CPU power model to being adaptive to VM environment.

Acknowledgment

This research work is partially supported by the National Natural Science Foundation of China (Grant Nos. 61772205 and 61402183), Science and Technology Planning Project of Guangdong Province (Grant Nos. 2017B010126002, 2017A010101008, 2017A010101014, 2017B090901061, 2016A010101007 and 2016B090918021), Guangzhou Science and Technology Projects (Grant Nos. 201607010048 and 201604010040).

Reference

[1] M. I. Green. Cloud computing and its contribution to climate change[J]. Greenpeace International, 2010.

[2] Data Center Users Group: Survey Results, tech. report, Data Center Users Group, Emerson Net- work Power, 2014.

[3] W. Lin, H. Wang, W. Wu. A Power Monitoring System based on a Multi-component Power Model. International Journal of Grid and High Performance Computing, 2018, 10(1):16-30.

[4] W. Lin, W. Wu. Energy Consumption Measurement and Management in Cloud Computing Environment[J]. Ruan Jian Xue Bao/ Journal of Software, 2016, 27(4): 1026-1041.

[5] E. O. Ofoegbu, E. Udoh. An Intelligent Power Load Control/Switching System Using an Energy Meter and Relay Circuit[J]. International Journal of Grid & High Performance Computing, 2016, 8(1):76-84.

[6] Massie M L, Chun B N, Culler D E. The ganglia distributed monitoring system: design, implementation, and experience[J]. Parallel Computing, 2004, 30(7):817-840.

[7] L. Luo, W. Wu, W. T. Tsai, D. Di, F. Zhang. Simulation of power consumption of cloud data centers[J]. Simulation Modelling Practice & Theory, 2013, 39(39):152-171.

[8] W. Barth. Nagios: System and Network Monitoring[M]. 2008.

[9] J. W. Smith, A. Khajeh-Hosseini, J. S. Ward, I. Sommerville. CloudMonitor: Profiling Power Usage[J]. 2012:947-948.

[10] A. E. H. Bohra, V. Chaudhary. VMeter: Power modelling for virtualized clouds[C]//Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW), 2010 IEEE International Symposium on. Ieee, 2010: 1-8.

[11] A. Kansal, F. Zhao, N. Kothari, and A. A. Bhattacharya. Virtual machine power metering and provisioning[J]. In: the 1st ACM Symposium on Cloud Computing (SoCC 2010), Indianapolis, Indiana, USA: ACM, 2011. 39-50.

[12] W. Lin, W. Wu, H. Wang, J. Z. Wang, C. H. Hsu. Experimental and quantitative analysis of server power model for cloud data centers[J]. Future Generation Computer Systems, 2016.

[13] T. Mudge. Power: A First-Class Architectural Design Constraint[J]. Computer, 2001, 34(4):52-58.

[14] Calculating Memory System Power for DDR3. https://www.micron.com/~/media/documents/products/technical-note/dram/tn41_01ddr3_power.pdf

[15] Standard Performance Evaluation Corporation. http://www.spec.org/power_ssj2008/results/,2016.

[16] J. Sievert. Iometer: The I/O performance analysis tool for servers[J]. 2004.

[17] C. H. Hsu, Poole S W. Power Signature Analysis of the SPECpower_ssj2008 Benchmark[C]// IEEE International Symposium on PERFORMANCE Analysis of Systems and Software. IEEE, 2011:227-236.

[18] R. Basmadjian, N. Ali, F. Niedermeier, H. De Meer, G. Giuliani. A methodology to predict the power consumption of servers in data centres[C]// ACM SIGCOMM, International Conference on Energy-Efficient Computing and NETWORKING. ACM, 2011:1-10.

[19] J. Janzen. Calculating Memory System Power for DDR SDRAM[J]. Designline Journal, 2001.

[20] Y. Kim, S. Gurumurthi, A. Sivasubramaniam. Understanding the performance-temperature interactions in disk I/O of server workloads[C]// The Twelfth International Symposium on High-Performance Computer Architecture. IEEE, 2006:176-186.

[21] M. Allalouf, Y. Arbitman, M. Factor, R. I. Kat, K. Meth, D. Naor. Storage modeling for power estimation[C]// of SYSTOR 2009: the Israeli Experimental Systems Conference 2009, Haifa, Israel, May. DBLP, 2009:1-10.

[22] D. Kliazovich, P. Bouvry, S. U. Khan. Simulation and performance analysis of data intensive and workload intensive cloud computing data centers[M]//Optical Interconnects for Future Data Center Networks. Springer New York, 2013: 47-63.

[23] W. Lin, W. Wang, W. Wu, X. Pang, B. Liu, Y Zhang. A Heuristic Task Scheduling Algorithm Based on Server Power Efficiency Model in Cloud Environments[J]. Sustainable Computing: Informatics and Systems, 2017.

[24] W. Lin, C. Zhu, J. Li, B. Liu, Huiqiong Lian. Novel algorithms and equivalence optimisation for resource allocation in cloud computing. IJWGS, 2015, 11(2): 193-210.

[25] W. Lin, S. Xu, J. Li, L. Xu, Z. Peng. Design and theoretical analysis of virtual machine placement algorithm based on peak workload characteristics. Soft Comput.,2017, 21(5): 1301-1314.

[26] W. Wu, W. Lin, Z. Peng. An Intelligent Power Consumption Model for Virtual Machines under CPU-intensive workload in Cloud Environment. Soft Computing.2017, 21(19):5755–5764.

[27] W. Lin, S. Xu, L. He, J. Li. Multi-Resource Scheduling and Power Simulation for Cloud Computing. Information Sciences, 2017, 397: 168-186.

[28] D. Xie, X. Lai, X. Lei, L. Fan. Cognitive multiuser energy harvesting decode-and-forward relaying system with direct links, IEEE Access, 2018, 6: 5596-5606.

[29] M. Z. A. Bhuiyan, J. Wu, G. Wang, Z. Chen, J. Chen, T. Wang. Quality-Guaranteed Event-Sensitive Data Collection and Monitoring in Vibration Sensor Networks, IEEE Transactions on Industrial Informatics, 2017,13(2):572-583.

[30] M. Z. A. Bhuiyan, J. Wu, G. Wang, and J. Cao. Sensing and Decision-making in Cyber-Physical Systems: The Case of Structural Health Monitoring, IEEE Transactions on Industrial Informatics, 2016,12(6): 2103-2114.

[31] W. Lin, Z. Wu, L. Lin, A. Wen, J. Li. An Ensemble Random Forest Algorithm for Insurance Big Data Analysis. IEEE Access, 2017, 5(11):16568-16575.

[32] L. Fan, X. Lei, N. Yang, T. Q. Duong, and G. K. Karagiannidis. Secrecy Cooperative Networks With Outdated Relay Selection Over Correlated Fading Channels, IEEE Trans. Vehicular Technology, 2017,66(8):7599-7603.

[33] U. Wajid, C. Cappiello, P. Plebani, et al. On Achieving Energy Efficiency and Reducing CO2 Footprint in Cloud Computing[J]. IEEE Transactions on Cloud Computing, 2016, 4(2):138-151.

[34] V. Adhinarayanan, W. C. Feng, D. Rogers, J. Ahrens, S. Pakin. Characterizing and Modeling Power and Energy for Extreme-Scale In-Situ Visualization[C]// Parallel and Distributed Processing Symposium. IEEE, 2017:978-987.

[35] T. Bostoen, S. Mullender, Y.Berbers. LofoSwitch: An online policy for concerted server and disk power control in content distribution networks[J]. Ad Hoc Networks, 2015, 25:606-621.

[36] J. Li, L. Sun, Q. Yan, Z. Li, W. Srisa-an, H.Ye. Significant permission identification for machine learning based android malware detection. In IEEE Transactions on Industrial Informatics. IEEE, DOI: 10.1109/TII.2017.2789219.

[37] Y. Li, G. Wang, L. Nie, Q. Wang. Distance Metric Optimization Driven Convolutional Neural Network for Age Invariant Face Recognition. Pattern Recognition. 2018, 75, 51–62.

[38] P. Li, J. Li, Z. Huang, T. Li, C.Z. Gao, S.M. Yiu, K. Chen. Multi-key privacy-preserving deep learning in cloud computing. Future Generation Computer Systems, 2017, 74:76-85.

[39] J. Li, Y. Li, X. Chen, P. Lee, W. Lou. A Hybrid Cloud Approach for Secure Authorized Deduplication. IEEE Transactions on Parallel and Distributed Systems, 2015, 26(5): 1206-1216.