A Preliminary Approach for Modeling Energy Efficiency for K-Means Clustering Applications in Data Centers

Da Qi Ren, Jianhuan Wen and Zhenya Li

Futurewei Technologies
2330 Central Expressway, Santa Clara, CA, 95050, USA
{daqi.ren, wenjianhuan, lizhenya}@huawei.com

Abstract. Energy efficiency is at the forefront of evaluating the performance of a data center in delivering green solutions for analyzing information in big data. A large scale distributed system is usually composed of a large number of power-hungry components. A methodology to model data processing flows inside the architecture of a data center is proposed to analyze the critical power constraints at the level of software. The model allows obtaining system characteristic values, thus benefiting analysts by providing the necessary environmental information to predict the power-efficiency alternative with the evaluation of energy consumption. An apparatus is designed comprising a receiver configured to receive a plurality of power measurements from a plurality of power sensors, and a processor coupled to the receiver and configured to determine an amount of power used by a processing element in a data center. The functionality and capability of this method for quantitative energy analysis of big data are validated by benchmarks and measurements performed on a real data center platform.

Keywords: Big Data, Energy Aware Computing, Performance modeling.

1 Introduction

Power dissipation is one of the crucial problems in the design of data intensive computing because big data applications need hundreds of hours of computations, consuming enormous amounts of energy. Large storage, many-core computing device and high speed network become the major choice in various big data processing platforms, the power consumption of such systems have been continually increasing. MapReduce/Hadoop has become one of the most important approaches in solving data intensive applications, where Hadoop Distributed File System (HDFS) works as the data storage mechanism and MapReduce as the computing engine. MapReduce/Hadoop does not change the nature of power consumption, however, much less research has been carried out to improve the

power performance with the integrated parallel programming paradigms in the architecture. Towards optimizing the power efficiency, we investigate software methodologies to analyze the power utilization through algorithm design and programming techniques.

Many algorithm level energy-aware design methods have been studied on parallel and distributed platforms. A typical approach in [1] introduces an integrated power model for a many-core computer to predict execution times and calculate dynamic power events. By integrating an analytical timing model and an empirical power model, the power consumption of workloads is predicted. Higher level models use more indirect and approximate design parameters, such as the algorithm level power model that we have introduced in [2], [3]. The advantage of the approach is that instruction mixture information, pipelining structure and out of order processing information can be covered in the data flows that are measured.

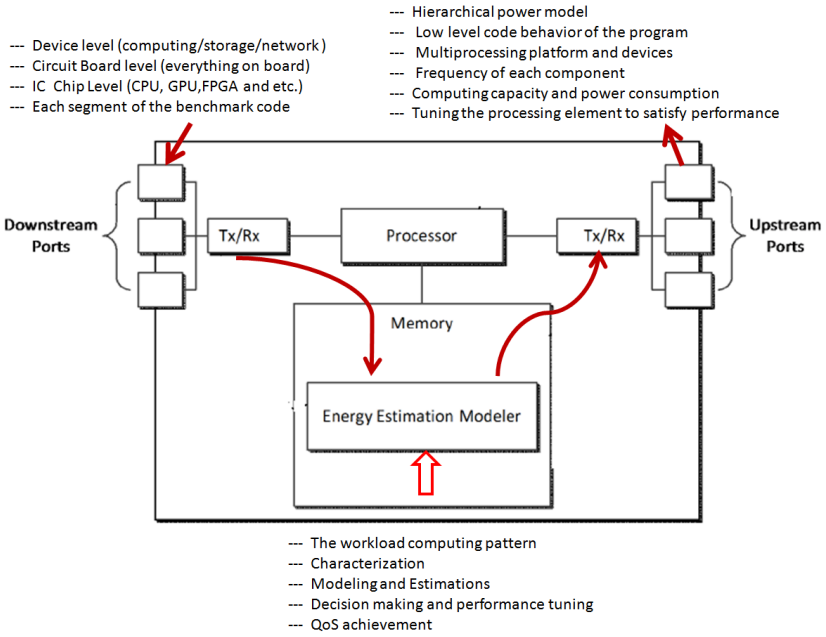

Fig. 1. An apparatus is designed to collect power usage data from data center, model the characterizations and tune the power efficiency.

2 Power Model

2.1 Power Measurement and Modeling

An apparatus is designed comprising a receiver configured to receive a plurality of power measurements from a plurality of power measurement instruments; and a processor with an energy measurement and estimation daemon coupled to the receiver and configured to: determine average power usage by a processing element in a data center by determining a summation of the plurality of power measurements; determine data to watt ratio that indicates power cost for the processing element to process that amount of data; determine the estimated execution time in a performance run for processing the amount of data by the processing element; and determine the estimated energy consumption that indicates the amount of energy to be used by the processing element to process the amount of data. In the apparatus shown in Fig.1, the plurality of power measurements are received from the plurality of power measurement instruments located at the device level; located at a circuit board level; and located at an integrated circuit level of a processing device in the data center. The data to watt ratio is determined by dividing data throughput of the processing element over a given time period by the amount of power used by the processing element over the given time period.

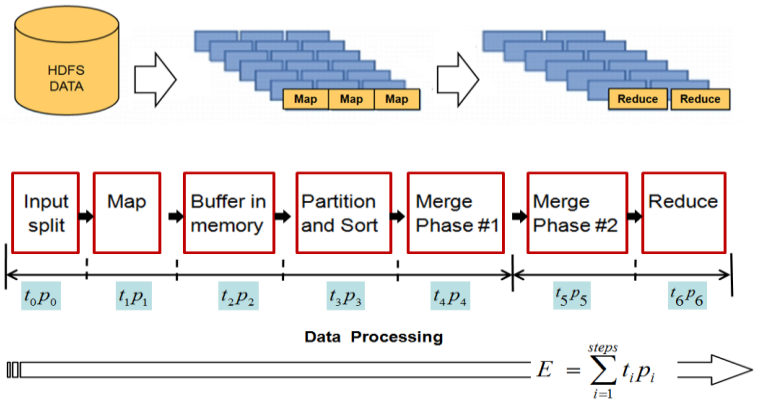

Fig. 2. A method for tuning performance of a processing element for handling data flow, comprises controlling the behavior of low-level code; abstracting and modeling the behavior of the program for analysis and study.

The estimated execution time is determined by dividing amount of data to be processed by the processing element by maximum data throughput of the processing element. the estimated energy consumption is determined by multiplying the estimated execution time by average amount of power used, determined according to the summation of the plurality of power measurements, wherein the apparatus is disposed within a data center system comprising a plurality of additional apparatuses, wherein the estimated energy for each of the additional apparatuses is determined in a manner similar to the apparatus, and wherein the estimated energy for the apparatus and the estimated energy for the additional apparatuses are aggregated and recorded by the data center.

As shown in Fig. 2, a method for tuning performance of a processing element for handling data flow is introduced in this work. It comprises controlling the behavior of low-level code of a program executing on the processing element; abstracting and modeling the behavior of the program for analysis and study; determining an estimate of energy used by the processing element in performing computational tasks; determining an overall power model for a multiprocessing platform; determining computational capacity of a parallel processing device; optimizing coding strategies according to the overall power model; and tuning the processing element to increase a power efficiency level of the processing element. In the method, determining the estimate of energy used by the processing element in performing computational tasks comprises multiplying an estimated execution time for executing the computational tasks by an average amount of power expected to be used in performing the computational tasks. The average amount of power expected to be used in performing the computational tasks is determined according to the amount of power used by the processing element while operating in a steady-state. The computational capacity of the parallel processing device is determined according to micro-processors of the parallel processing device, programming language used in the parallel processing device, and characteristics of the computation performed on the parallel processing device. Tuning the processing element to increase power efficiency level comprises at least one of domain partitioning, load parallelization, dynamic frequency scaling, or workload scheduling. The overall power model is determined according to the accumulation of power measurements for each component in the processing element. The method operates in both time domain and at hardware domain, frequency of each component is varied, computing capacity and power consumption vary according to the frequency of each component, and wherein the frequency of each component is varied to tune the processing element to satisfy the performance requirement. The method facilitates prediction of power usage, and comprises energy model that is based on workload characteristics of the processing element.

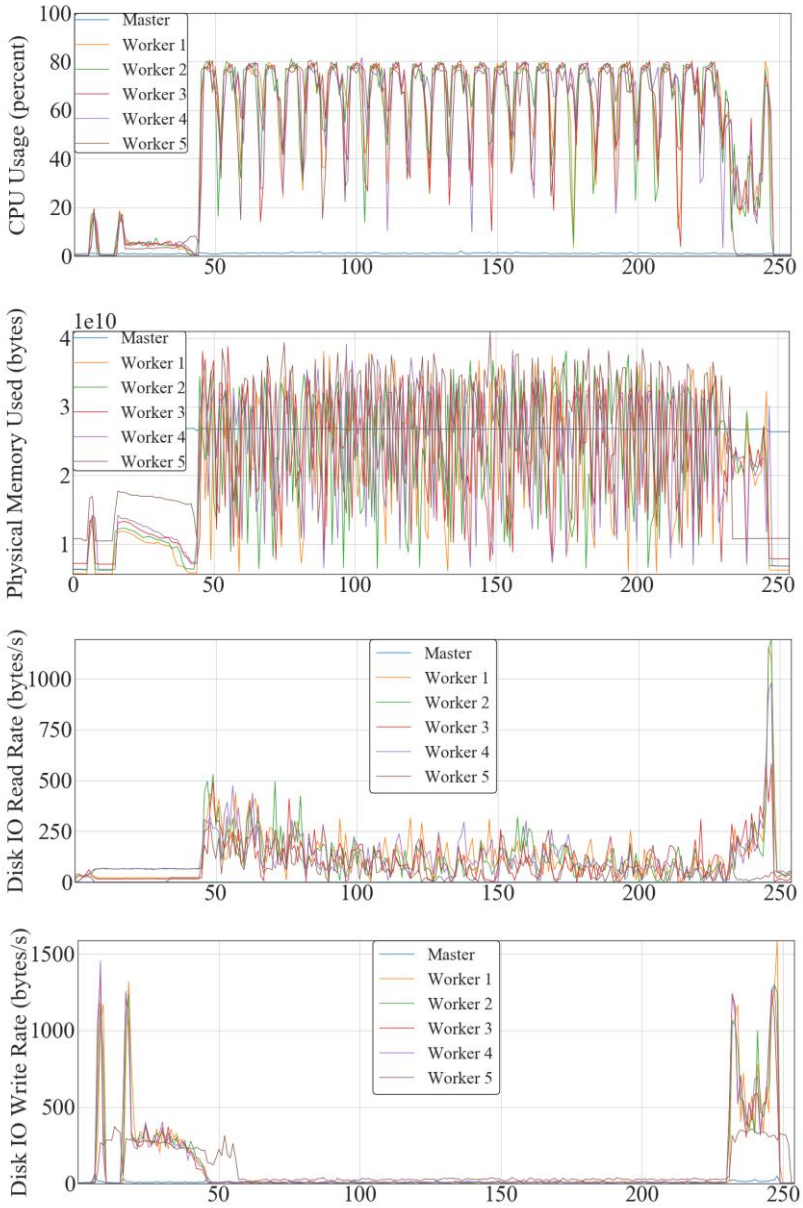

Fig. 3. Combined CPU usage, physical memory usage, disk IO read and write rate for all 6 nodes in the cluster, when executing K-means clustering with iterations I = 25.

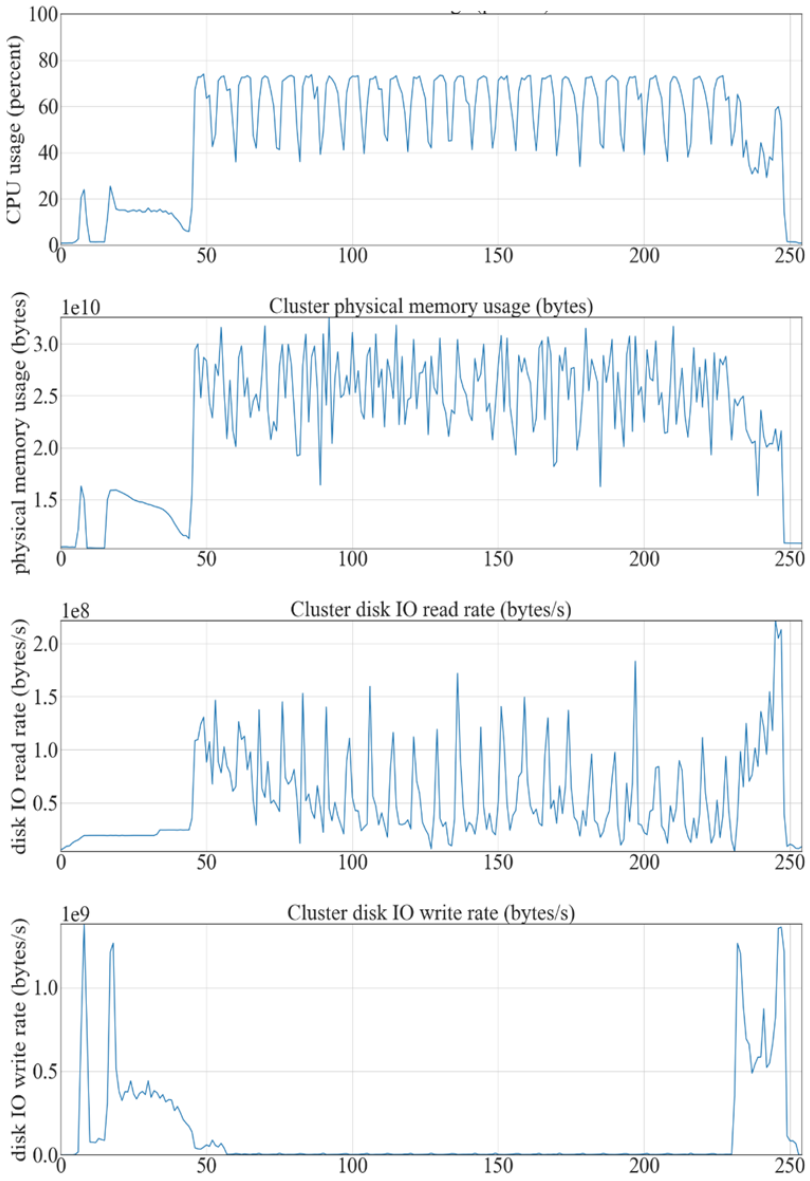

Fig. 4. Combined CPU usage, physical memory usage, disk IO read and write rate for the cluster, when executing K-means clustering with iterations I = 25.

2.2 Energy Consumption Characteristics

An application can be modeled based on the computing patterns and characteristics of its workload. The energy approximation is the summation of the products of each component power and its execution time [2]. An overall power model can be built for the entire multi-processing platform based on the same. Computation capability of a parallel processing device, e.g. many-core compute device and storage device, is determined by its micro-architecture, programming language and characteristics of the computation performed on it [4], [5]. The methodology imports hardware power parameters to the software algorithm study, then estimates power consumption with program analysis. One of the advantages is that it allows obtaining design characteristic values at the early programming stage, thus benefiting programmers by providing necessary environment information for choosing the best power-efficient alternative.

3 Workload Analysis

3.1 K-means Clustering

K-means clustering is one of the most popular flat clustering algorithms and can be configured to match the complexity of real-world use cases [5]. It is an unsupervised clustering technique comprising the following steps: Given total objects n, and a number of clusters k, repeat calculating and regrouping the n objects into k (≤ n) sets so as to minimize the sum of squares (WCSS) (i.e. variance) within each cluster. The algorithm is repeated until desired convergence level is achieved.

3.2 Power Measurement

Energy is consumed by each device in a System Under Test (SUT). A total power is the summation of each power source $P = \sum_{1 \le i \le m} p_i$, where m is the total number of devices, p_i is the power measurement of each device i. In real measurement, power results in each sampling period can be plotted together to obtain a power chart for the program, and the chart shows the power usage against time during the execution. For each device or subsystem, the calculation defined for Energy metrics is $E = \int_0^T P(t)\, dt$, where T is the elapsed time for the performance run, $P(t)$ is the power measured at time t.

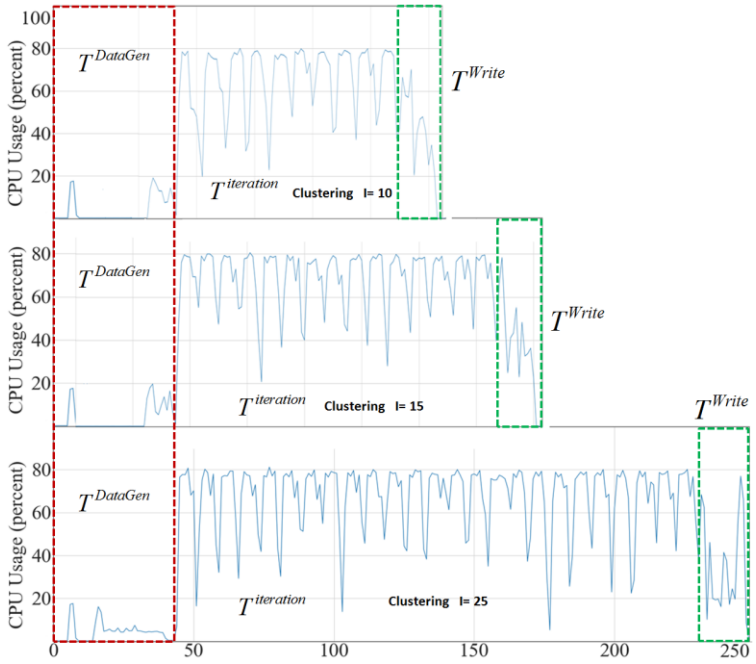

Fig. 5. CPU usage for the cluster, when executing K-means clustering with iterations I=10; I=15; and I=25.

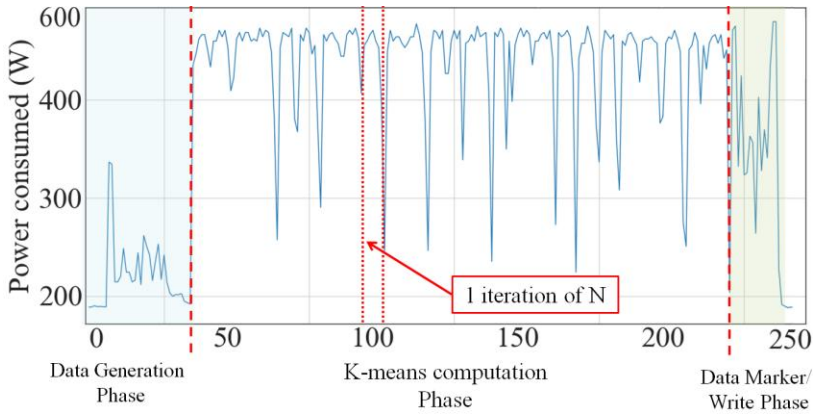

Fig. 6. Energy consumption model of K-means clustering with three phases.

Table 1. The experiment results by compared the estimated energy consumption and the real measurement values.

		Energy consumption of K-means Cluster			
		Phases	***Estimation(J)***	***Measurement(J)***	***Errors***
K=10	**CPU**	Generation	407,971.00		0.000%
		Clustering	1,712,980.00	1,867,599.00	9.026%
		Writeback	299,213.00		0.000%
		Overall	2,420,164.00	2,574,783.00	6.389%
	Server	Generation	408,879.00		0.000%
		Clustering	2,108,040.00	2,298,417.00	9.031%
		Writeback	299,176.00		0.000%
		Overall	2,816,095.00	3,006,472.00	6.760%
K=15	**CPU**	Generation	408,031.00		0.000%
		Clustering	2,644,470.00	2,851,399.00	7.300%
		Writeback	298,970.00		0.000%
		Overall	3,351,471.00	3,558,400.00	5.800%
	Server	Generation	409,612.00		0.000%
		Clustering	3,162,060.00	3,282,625.00	3.813%
		Writeback	299,891.00		0.000%
		Overall	3,871,563.00	3,992,128.00	3.114%
K=25	**CPU**	Generation	407,280.00		0.000%
		Clustering	4,512,413.00	4,718,772.00	4.573%
		Writeback	299,170.00		0.000%
		Overall	5,218,863.00	5,425,222.00	3.954%
	Server	Generation	408,311.00		0.000%
		Clustering	5,130,147.00	5,411,022.00	5.200%
		Writeback	299,771.00		0.000%
		Overall	5,838,229.00	6,119,104.00	4.600%

3.3 Program Segments

The K-means clustering computation includes three major phases as the follows [5]: 1) data generation phase: to generate random object data and store the data in storage; 2) clustering phase: to drop outliers, scale observations, distance measure, clustering, adjust parameters, repeat multiple iterations and finally choose representative components; 3) writing phase: marking data with the cluster information and write the data back to storage. Let $T^{DataGen}$ represents the time for data generation in phase 1; $T^{iteration}$ represents the time for clustering computation in phase 2; and $T^{Writeback}$ represents the time to write the data in phase 3. The estimated energy consumption for completing the program is the average power multiplied by the execution time, as shown in Equation (1).

$$E_{estimation} = \sum P_{AveragePower}^{phase} \times T^{phase} \qquad (1)$$

where E is the energy estimation, $P_{AveragePower}^{phase}$ is the average power consumption while the application run in each steps, T^{phase} is the computation time for the K-means clustering phases of data generation, clustering and writing.

3.4 Equations and Measurements

Power consumption of a computing platform is measured or calculated for each node separately. The total power consumption can be modeled as

$$P(w) = \sum_{i=1}^{N} P_{Compute}^{i}(w^{i}) + \sum_{j=1}^{M} P_{Storage}(w^{j}) + P_{Network}(w) \quad (2)$$

where P, $P_{Compute}$, $P_{Storage}$ and $P_{Network}$ represent the power of the total, computing device, storage device and network devices, respectively. N and M are the numbers of computing devices and storage devices involved in the computation of workload w. w^{i} and w^{j} represent the workload assigned to $Compute\_device_{i}$ and $Compute\_device_{j}$ respectively. The power consumption of computing device is dominant among all the entities consuming power in each node. The analysis and prediction of the power consumption of a Hadoop cluster are introduced below, by using the benchmark results of *K*-means clustering.

3.5 Power Model

While data size and a computing platform and configuration are fixed, the time for data generation $T^{DataGen}$ will take an approximate identical time in each run of K-mean clustering on the platform. According to the conditions of convergence and the accuracy threshold setup, a K-means clustering may take a different number of iterations to complete. Based on the algorithm, the time to complete each iteration $T^{iteration}$ can be considered as a fixed value [5]. After finished the last iteration, the time to write the data back $T^{Writeback}$ can be assumed as the same since the data size will not be changed. Assuming the total number of iterations is *k*, the total computing time can be represented in Equation (3).

$$T_{Total} = T^{DaraGen} + kT^{iteration} + T^{write} \qquad (3)$$

And the total energy consumption can be represented in Equation (4).

$$E_{Total} = E^{DaraGen} + kE^{iteration} + E^{write} \qquad (4)$$

4 Performance Results

4.1 Power Consuming Experiments

For the experiments in this work, we use six HUAWEI Tecal RH2288 V2 Rack servers, each with 2 Intel Xeon Processor E5-2680 (Sandy Bridge-EP) running at 2.7 GHz [6]. Each processor has 8 cores (16 hyperthreaded) and a 20MB L3 cache. They are connected through two QuickPath links, each providing a unidirectional transmission rate of up to 8.0 GT/s. Each server has 24 8GB double data rate 3 (DDR3) at 1066Mhz dimms of main memory with a total ot 192GB. Each server is configured with eight 2.5" SAS HDDs with 7.2TB capacity. One SAS disk hosts the OS and the remaining 7 are configured for HDFS. The server provides four onboard gigabit Ethernet (GE) ports, of which two were bonded to double bandwidth. Our system runs CDH 5 on centos 6.5. Cloudera is configured as one master node and five worker nodes. Each of the nodes has 189.1GBytes of memory. We use Apache Hadoop 2.6 and open source HiBench K-means that is popular for processing large datasets built for distributed applications. The power analyzer used in this test is HIOKI 3600.

4.2 Power Consuming Characterization

The low level processing is finally turned to be a sparse metrics production to computer the vector distances. It makes the computing devices to execute the input data follow inside each K iteration Single Instruction Multiple Data (SIMD), i.e. multiply and plus operations repeatedly in a streaming way. Therefore, when the data size, the processor's frequency and temperature are invariant, the CPU power can be modeled as a constant value inside each K-iteration.

The resource usage and the consumption of K-means clustering computations for 25 iterations are shown in Fig.3 and Fig. 4 for the whole cluster and one single node, respectively. This status is true in all Hadoop nodes, the total energy used to complete the K-means clustering task is linearly related to the input data size, I iterations and K number of clusters. For experimental purposes, a random sample of 1.2 billion records with 20 variables was used. Power characterization and stress testing was done by varying the number of K for a fix data set of 225GB.

5 Energy and Estimation

5.1 CPU Usage and CPU Power

For estimating the relationships between the CPU usage and the energy consumption, we first plotted the CPU usage and power against time. Visual inspection indicates a strong relationship between CPU usage and power. For the SIMD character of this experiment, the relationship is considered to be linear, allowing us to use statistical methods to predict power by CPU usage.

5.2 Workload Characters, Performance and Power

Fig. 5 shows the computation steps include data generation; clustering phase; and data marking and writing back for K-mean iteration I=10, I=15 and I=25, respectively. Fig. 6 shows the energy measurement of one worker node for K-mean iteration I=25. While the total data size is identical, the time for data generation can be fixed as a constant, therefore the corresponding energy consumption is determined. The clustering computation follows the SIMD rule, the time taken is linear to the number of computing iterations. If a single iteration takes time T and consumes energy E, the total energy consumption during K-means with I=25 can be modeled as by one E energy multiplied by number of iterations, i.e. 25E. Because the number of objects N will not change at the marking and writing back stage, the corresponding time and energy consumption can be identically determined. A detailed energy consumption is listed in Table 1.

6 Error Analysis

Power modeling and estimation at software level is relative accurate because of the integration between resource usage, the platform and physical measurement. However, the errors in power estimation of K-means clustering algorithm lies in the throughput to power ratio displacement. It causes the power values to not precisely fix to the same throughput when the workload size is small. This can be seen in the measurement of the peak dynamic power where the curve is not a straight line. The K-means R/W overheads and under-foots may also slightly change at each time when a new iteration of clustering launches. The power phase leg to the assembly execution will cause the calculation error, especially when the data size is small.

The clustering phase of K-means has two stages, namely iteration (CPU-bound) and regrouping (I/O-bound). On the platform, CPU dominate the overall power consumption. The overhead introduced by storage devices, network devices are negligible compared with CPU. A linear model was developed to predict the power consumption based on CPU usage. The results of modeling are shown in Table 1. The energy consumption during clustering was calculated in multiple ways. The energy can simply be calculated from power using area under the curve. From Table.1, when iteration number I=10 the energy used by one server node during actual computation was 2,816,095 J, while the same energy calculated from the model is found to be 3,006,472 J, with an error of 6.76%. When iteration number I=15 and I =25, the energy used by one server node estimated using the model is found with an error of 3.11% and 4.60% compared with the real measurement values, respectively. All calculations verify the results despite having synchronization between measurements and process runtime. This is significant because it allows a programmer to tune the power efficiency of an application by simply tune the CPU usage such as frequency, etc.

7 Conclusion and Future Work

A general approach is introduced for quantitative power estimation and analysis on Hadoop clusters for big data computing. The workload characteristics have been analyzed. Based on this, power parameters are captured by measuring the power of each component in a PE on a real system. The Hadoop PE's power feature in executing the K-means clustering program can then be analyzed and concluded. The power consumption of a K-means program with same computational characteristics of any size that is running on the PE can be predicted based on the power consumption feature. Finally the approach is validated by measuring real computation power on the target hardware. The power analysis method can be refined to enhance its precision by including more components, and based on it the power parameters can be tuned for obtaining the best power performance for given problems. These will be considered in our future work.

Acknowledgement

We thank Mr. Chaitanya Kundety for his assistance with experiments and the comments that improved the manuscript.

References

1. Sunpyo Hong, Hyesoon Kim, "An integrated GPU power and performance model", International Symposiumon Computer Architecture, Saint-Malo, France, (2010).
2. H. Yang, Q. Zhao, Z. Luan, Depei Qian: iMeter: An integrated VM power model based on performance profiling. In: Future Generation Computer Systems, Volume 36, July 2014, Pages 267–286.
3. Da Qi Ren: Algorithm level power efficiency optimization for CPU–GPU processing element in data intensive SIMD/SPMD computing. In: Journal of Parallel and Distributed Computing 71(2):245-253, Feb 2011, DOI: 10.1016/j.jpdc.2010.10.007.
4. M. Malik, S. Rafatirah, A. Sasan1, H. Homayoun1: System and Architecture Level Characterization of Big Data Applications on Big and Little Core Server Architectures. In: 2015 IEEE International Conference on Big Data, Santa Clara, USA, Oct 29-Nov 1, 2015.
5. Tou, J. T., and Gonzalez, R. C: Pattern Recognition Principles. In: Addison-Wesley; 2nd edition (August 1977), ISBN-10: 0201075873, ISBN-13: 978-0201075878
6. Intel® Xeon®Processor E5 v3 Product Family, Processor Specification Update, Sep 2016. https://www.intel.com/content/dam/www/public/us/en/documents/specification-updates/xeon-e5-v3-spec-update.pdf .