DV-DVFS: Merging Data variety and DVFS Technique to Manage the Energy Consumption of Big Data Processing

HOSSEIN AHMADVAND, Sharif University of Technology, Tehran, Iran
FOUZHAN FOROUTAN, Sharif University of Technology, Tehran, Iran
MAHMOOD FATHY, Iran University of Science and Technology and School of Computer Science, Institute for Research in Fundamental Sciences, Tehran, Iran

Email address of the corresponding author: ahmadvand@ce.sharif.edu

Abstract. Data variety is one of the most important features of Big Data. Data variety is the result of aggregating data from multiple sources and uneven distribution of data. This feature of Big Data causes high variation in consumption of processing resources such as CPU consumption. In this paper, we used Dynamic Voltage and Frequency Scaling (DVFS) to reduce the energy consumption of computation. To this goal, we consider a deadline as our constraint and before applying the DVFS technique to computer nodes, we estimate the processing time and the frequency needed to meet the deadline. We have used a set of data sets and applications in the evaluation phase. The experimental results show that our proposed approach surpasses the other scenarios in processing real datasets. Based on the experimental results in this paper, DV-DVFS can achieve up to 15% improvement in energy consumption.

KEYWORDS: Data variety, Data Skew, DVFS, Big Data

1. INTRODUCTION

In recent years, due to the huge amount of generated data and high volume of processing capacity, the power and energy management of this huge processing, is a considerable value. The authors in [1] estimate that over 40% of the budget in a data center has been spent on electrical power and cooling in 2008. Due to the structure of MapReduce processing and 4'v of Big Data, Big Data processing is a suitable area to apply the power reduction techniques such as DVFS. Furthermore, as we shown in the previous work [2] data variety that is one of the important features of big data causes variation in resource consumption. This fact makes DVFS a suitable technique for reduction of power/energy consumption in big data processing.

There are several approaches that addressed this issue. Some recent works considered the energy consumption in MapReduce-like distributed processing frameworks [3], [4]. The goal of these researches was to determine the number of worker nodes used by a Hadoop cluster to minimize energy consumption and at the same time guarantee a specified deadline. However, they assumed that they know the input data beforehand. This assumption is not valid in a real environment, so the applicability of such techniques is limited. Furthermore, there are multiple schedulers that consider the deadline constraints [5]; nevertheless, none of the schedulers supports the minimization of energy consumption. The authors in [6] have presented a framework for Efficient Energy Scheduling of Spark workloads. The authors in [7] have used load balancing to improve energy efficiency. They have used the heuristic method for their goal.

Data variety causes some variation in generating results and resources utilization. None of the previous works has paid attention to this issue. To address this challenge; we present our power- conscious approach to manage the energy consumption of big data processing. As Fig. 1 shows, we use sampling to discover the input data. We have used pre-processing and an estimator to estimate the frequency and time of processing.

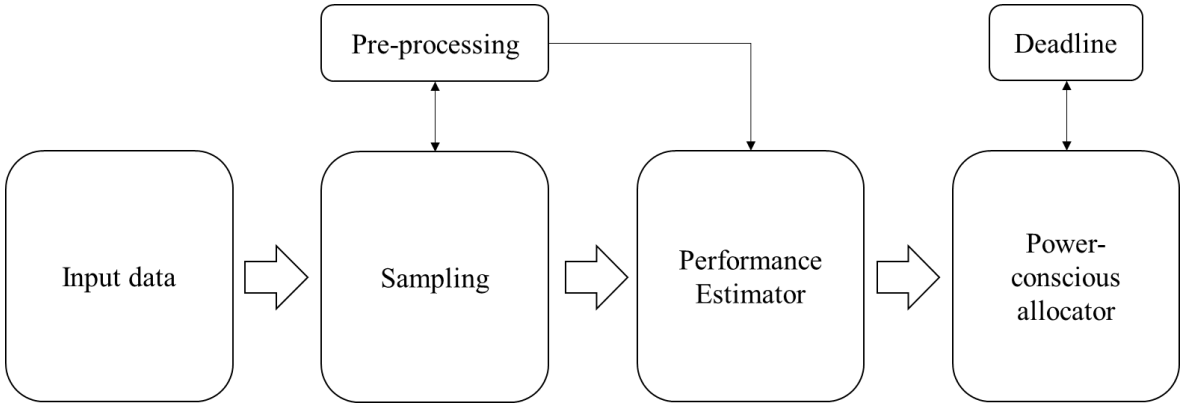

Fig. 1. our approach

Contributions. In this paper, we have the following contributions:

1. We have presented a framework to consider data variety/ skew for efficiently assigning resources in Big Data Processing.
2. We have used sampling to discover the amount of data skew.
3. We have implemented our approach in Spark environment and evaluated it by some well-known datasets and applications.

Organization. The rest of the paper is organized as follows: Subsection 2 presents the motivation of our work. Section 3 presents an overview of the state of the arts and previous works. Section 4 describes the proposed approach and system design. The experimental result and evaluations are presented in section 5, and Finally, Section 6 includes the main conclusions and future works.

2. MOTIVATION

As discussed in the previous section, there is an opportunity to reduce the energy consumption in Big Data processing. To show this opportunity, we aggregate 23 GB data from four sources and divide it into 0.5GB blocks. We consider these blocks and show the average CPU utilization and processing time of them. Fig. 2 and Fig. 3 shows average CPU utilization and processing time for various applications and different parts of input data. Experiments of the current section were run on an Intel Core-i7 4-core CPU at 2.8GHz with 4GB of RAM.

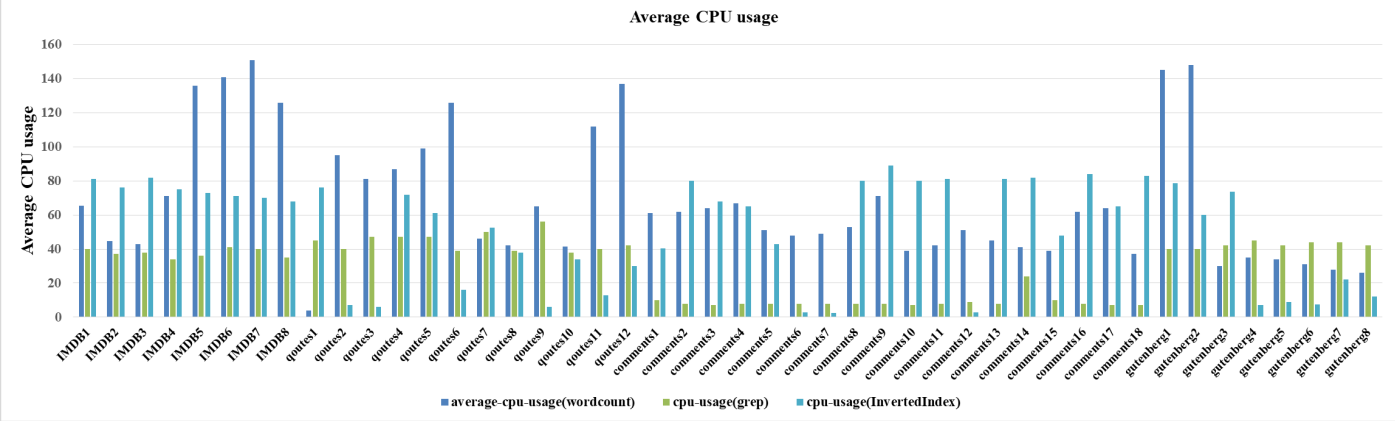

Fig. 2. UPU utilization in various parts of sources

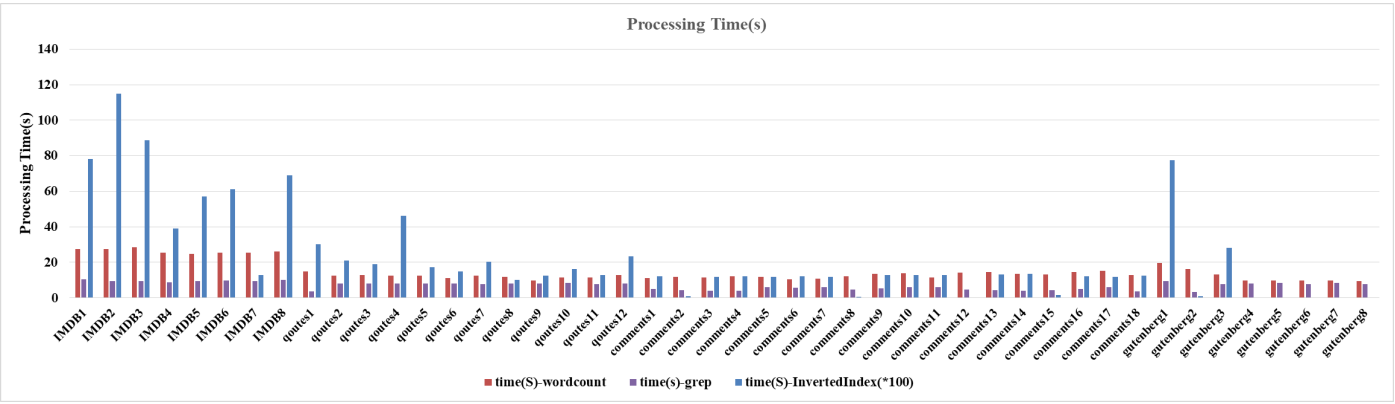

Fig. 3. Processing time for various parts of sources

Table 1, presents the average, variance and coefficient of variation of CPU usages and processing time in each benchmark.

Table 1. CPU utilization in three applications

Applications	WordCount	Grep	Inverted Index
Average-CPU usage	68%	45%	82%
Variance CPU usage	42	17.5	30.5
C.V of CPU usage	0.51	0.65	0.64
Average-average Processing time	14.9	6.97	2283.5
Variance of Processing Time	31.3	4.12	6881847.7
C.V of Processing Time	2.1	0.6	3013.8

As shown in Table 1, there is a significant opportunity to manage CPU utilization and power consumption. We focused on this fact in this paper. So, we used DVFS technique for processing of some parts of input data.

Based on the presented content, we should answer the following question:

Why using DVFS in big data processing?

1. Data variety causes a significant diversity in resource utilization.
2. Using cloud computing for big data processing intensified the data variety and causes more skew in data.
3. Input data usually aggregate from multiple sources. This fact also intensifies the skew of data.
4. MapReduce is a well-known paradigm for big data processing. MapReduce is consisting of two main parts: Map and Reduce. Each phase of this paradigm has various impact on the utilization of resources.

3. RELATED WORKS

Related works of our research are divided into 2 main categories. We have presented these categories in Fig. 4. The categories are:

1. Using Dynamic Voltage and Frequency Scaling (DVFS) to energy reduction.

2. Another approach to reduce energy consumption.

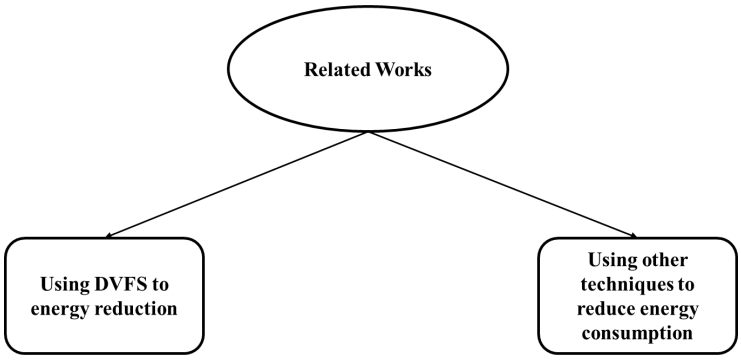

Fig. 4. The categories of related works

Using DVFS to energy reduction. DVFS is the well-known approach to reduce energy reduction in case of the lack of energy. The authors in [6] used DFVS to reduce the energy consumption of MapReduce applications. They have compared their work with the default Spark Scheduler. VM migration and scale down in case of low performance is considered in [8]. The authors in [9] have considered the variation of applications requirements in Big Data in case of choosing cloud as an infrastructure for processing. Tuning CPU frequency based on the QoS has presented in [10]. They have used a prediction method for adapting the frequency depends on the QoS and available slack. They have reduced the energy consumption of heterogeneous Hadoop Cluster. The authors in [11] have used DVFS and machine learning approaches to reduce energy consumption in NoCs. The authors in [12] have used DVFS for microprocessors power and energy reduction. DVFS based policies are used in [13] for consolidation of virtual machines for energy-efficient cloud data centers. The authors in [14] have

merged a thermal-aware approach and DVFS to manage the energy of data center. In this paper, the factors of energy inefficiency are divided into two categories: resources underutilization and heat effects.

Using other techniques to reduce energy consumption. The authors in [15] considered the server utilization to reduce energy consumption. They have considered QoS in their problem. The authors in [16] have used Data-driven approach to improve the performance of HPC systems. The authors in [7] have presented a heuristic-based framework for energy reduction by load balancing. The authors in [3] and [4] have considered the energy consumption in MapReduce like distributed processing frameworks. The goal of these researches was to minimize energy consumption and guarantee deadline by determining the number of worker nodes.
The impacts of failures and resource heterogeneity on the power consumption and performance of IaaS clouds have been investigated in [17]. The authors have used historical monitoring data from the online analysis of the host and network utilization without any pre-knowledge of workloads for the reduction of SLA-violation and energy reduction in [18].
The authors in [19] have considered application-level requirements for energy reduction. They have considered the effect of the variety of workload on the utilization of VMs and network. They have reduced the energy cost by assigning the suitable amount of resources to the VMs. The authors in [20] survey the previous works on energy consumption of datacenters. They have divided the research areas to some parts and discussed them. They have not considered the data variety/ skew in their study.
We have also considered data variety and reduced the processing resources such as energy or cost [21], [22].

4. Methods

In this section, we present problem definition and the algorithm of the proposed approach. We have considered data variety/skew in data for this method. Our problem is the reduction of energy consumption by applying DVFS to the computer nodes to overcome the inefficiency causes by data skew. For this reason, we divide the input data into some same size portions. We estimate the required processing resources for each portion by using sampling. Then, we select the suitable portions for applying DVFS techniques. In this problem, we must consider the deadline as a constraint. We have used the DVFS technique to reduce energy consumption and meet the deadline.

For solving this problem, we have presented a heuristic approach. In this heuristic approach, we have used some notation for our presentation of the problem. Table 2 presents the notations that we used.

Table 2. Notations that used in this paper

Notation	Description
D	Deadline
EC	Energy Consumption
FT	Finish time
UF	Utilize Factor
TS	Time Slot
B_i	The i-th block
PT_i	The processing time of i-th block
RPC	Required Power for Processing
REP	Required Energy for Processing
SFB_i	*Suitable Frequency for processing* B_i

Problem Statement. EC presents the energy consumption in this paper. We try to minimize the EC while the deadline should be met. So, the deadline is the constraint of our problem.

Problem formulation. The objective function to be minimized is the energy consumption and the constraint is the deadline.

$$Min(EC) \qquad (1)$$

Subject to:

$$FT \leq D \qquad (2)$$

(1) Presents the objective function and the (2) presents the constraint of our work.

To overcome the above problem, we have presented algorithm1. Before the presentation of the algorithm, we define a parameter "Utilize Factor".

$$P_i = (P_i^{full} - P_i^{idle}) * u_i^{CPU} + P_{idle} \qquad (3)$$

$$u_i^{CPU} = UF_i * u_i^{full} \qquad (4)$$

$$UF_i = PT_i / TS_i \qquad (5)$$

$$\sum_{0}^{N} TS_i \leq Deadline \qquad (6)$$

$$EC = \sum_{0}^{N} PT_i * P_i \qquad (7)$$

Formula 3 to 6 calculate the RPC for each block. Formula 6 presents the constraint of the problem. Formula 7 calculates the energy consumption of processing.

Our algorithm. Our algorithm is presented below.

Algorithm1

1: ***Input:*** *Deadline,*
2: ***output:*** *SFB_i*
3: ***divide*** *(Deadline into N_{DP} Parts)*
4: ***while*** *(PT < Deadline)*
5: ***Sample*** *(initial Blocks)* *// to discover the amount of skew*
6: ***Estimate*** *(SFB_i)*
7: ***end while***

Lines 1-2 of the algorithm1 is initializing the variables. Note that *REP* and *PT* show the required energy and time for processing. Line 3 divides deadline into some time slot. Line 5 uses sampling to discover the skew in initial blocks. By using this sampling method, we able to estimate the amount of required resources for processing for each block. Line 6 estimates the suitable frequency for processing of B_i. For this issue, finishing time of processing should be lower than the time slot.

Complexity Analysis. The algorithm time complexity is of O ($\log_2$ (n)) where n represents the volume of input data. Line 4-6 of the algorithm is a loop procedure and takes O ($\log_2$ (n)) of time.

Implementation. In our approach, we divided the input data into some data blocks. In Spark environment, these blocks are converted into some RDDs[1]. We have used sampling to discover the amount of processing resources needed for processing each RDD. Based on this information we have decided the amount of resources needed for processing each RDD. As Fig. 5. shows, certain frequency is assigned to each RDD. So, by applying this approach, we have used dynamic voltage and frequency scaling for big data processing.

[1] Resilient Distributed Datasets

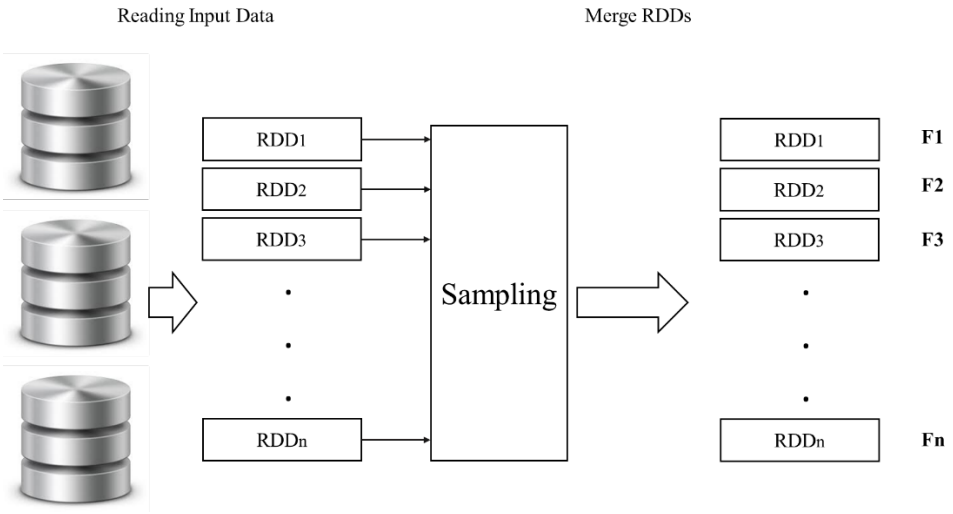

Fig. 5. Overview of our approach

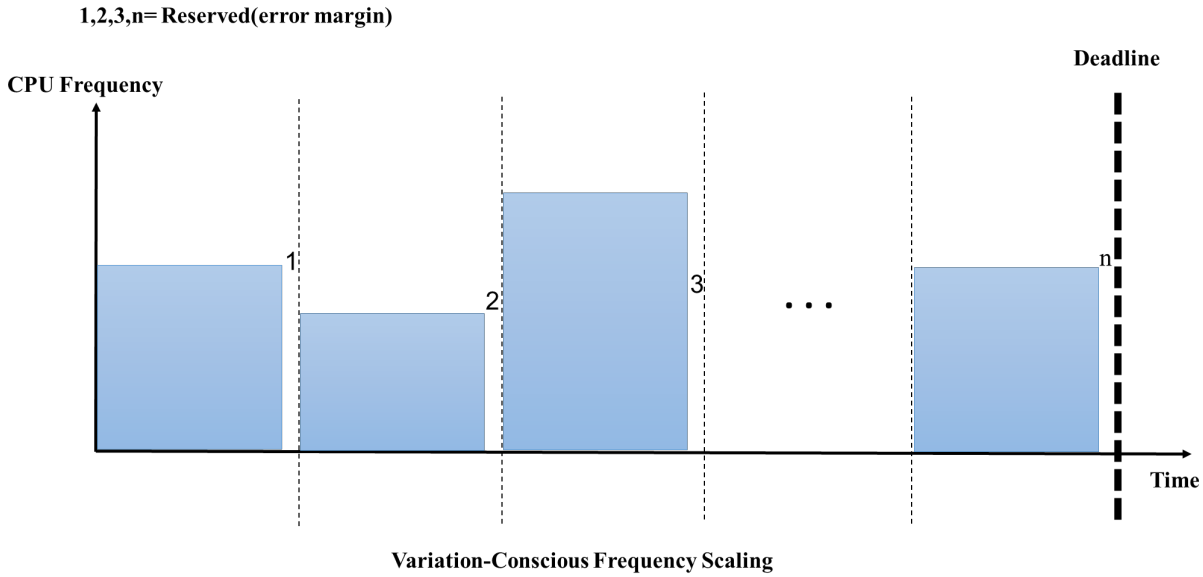

Fig. 6. Variation-Conscious Frequency Scaling (our approach)

Fig. 6. shows that the DVFS is applied to RDDs and we decide the resources in which the deadline can be met. As Fig. 6. shows in each time slot, we have considered a reserved area for error margin.

5. RESULTS and DISCUSSION

We used three benchmarks from BigDataBench suite [23] in our evaluation process. We also have used TPC Benchmark (MAIL, SHIP, AIR, RAIL, TRUCK) and Amazon review dataset (Music, Books, Movies, Clothing, Phones) [24], [25]. Amazon product data contains product reviews and metadata from Amazon, including 142.8 million reviews spanning May 1996 - July 2014. TPC-H is a decision support benchmark. It consists of a suite of business-oriented ad-hoc queries and concurrent data modifications. The queries and the data populating the database have been chosen to have broad industry-wide relevance. This benchmark illustrates decision support systems that examine large volumes of data, execute queries with a high degree of complexity, and give answers to critical business questions. We have used four different sources [26], [27], [28] and Wikipedia for WordCount, Grep, Inverted Index, and AverageLength. We have used a bootstrapping method for generating 100GB data as input datasets [29]. Experiments were run on three machines, Intel Core-i7 4-core CPU at 2.8 GHz with 4 GB of RAM. We apply the DVFS to some parts of data and reduce CPU frequency to the 1.6 GHz.

Applications. Applications are as follows:

- **WordCount**: This application Counts the number of words in the file.
- **Grep**: It searches and counts a pattern in a file.
- **Inverted Index**: This application is an index data structure storing a mapping from content to its locations in a database file.
- We also consider **AVG** for TPC-H datasets and **SUM** for Amazon datasets.

Comparison. We have compared our approach with a default scheduler of Spark (FAIR).

FAIR. The same amount of resources is given to each application. In this kind of frequency scaling, we have considered a fixed frequency as CPU frequency (i.e., default Spark scheduler).

Fig. 7 to Fig. 11 depict the execution time and energy consumption of approaches. As it can be seen, our proposed approach can surpass the other in all the applications in term of energy consumption. Based on the deadline as a constraint, we have delayed the completion of processing. We have met the deadline in all applications.

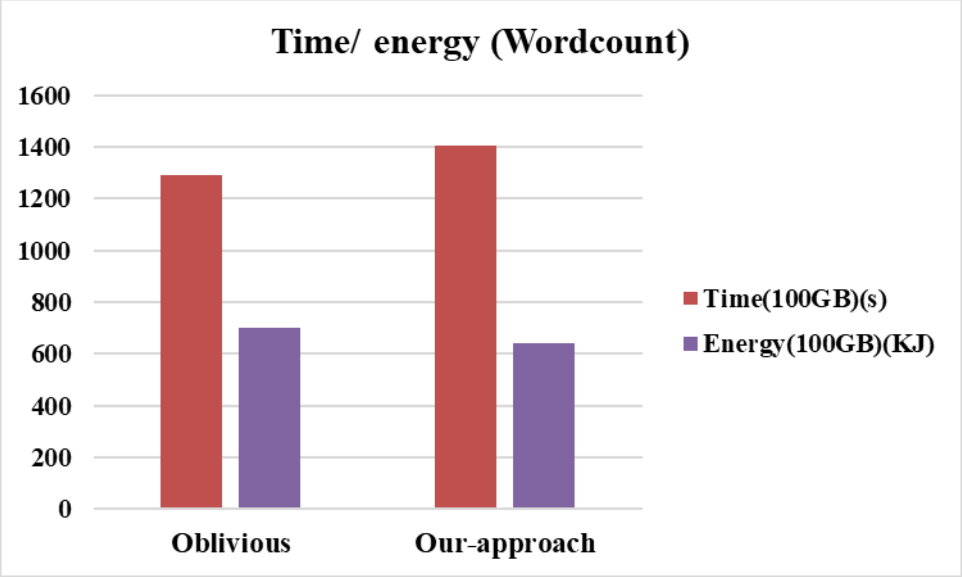

Fig. 7. Processing time and energy of WordCount

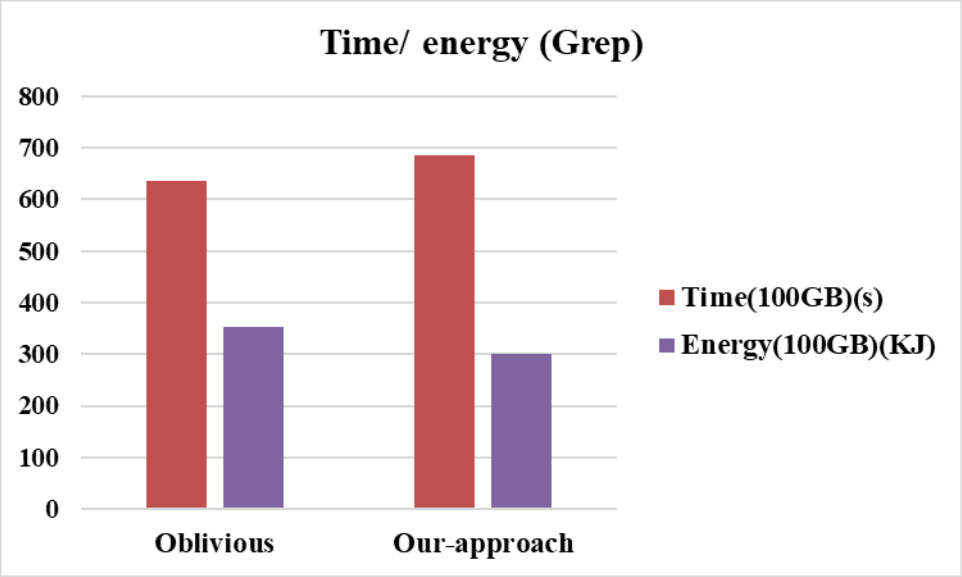

Fig. 8. Processing time and energy of Grep

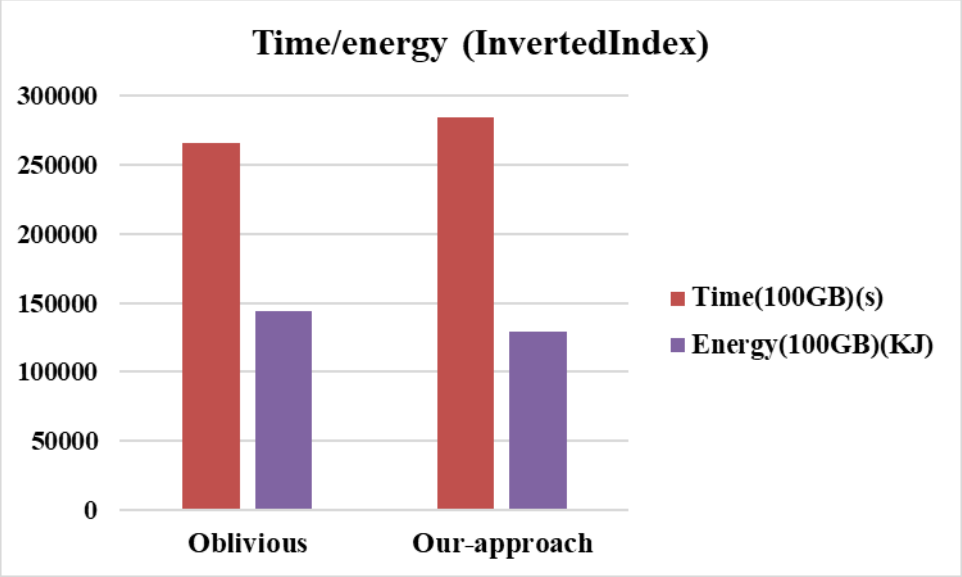

Fig. 9. Processing time and energy of Inverted Index

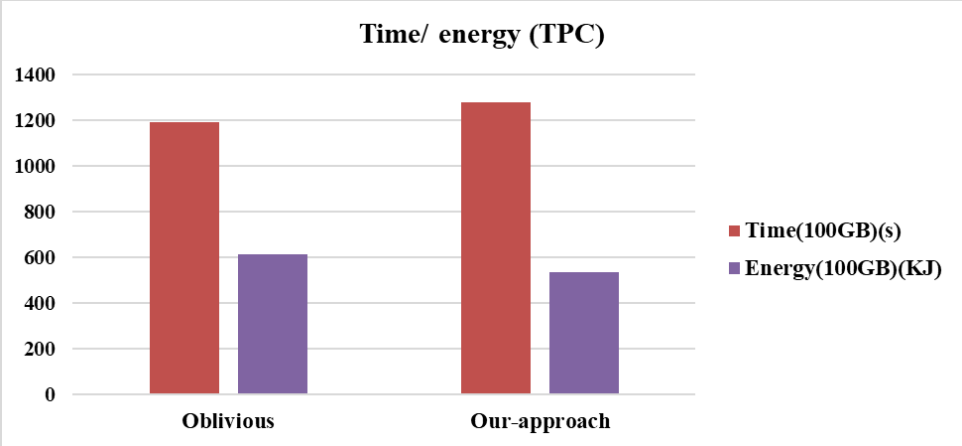

Fig. 10. Processing time and energy of TPC Datasets

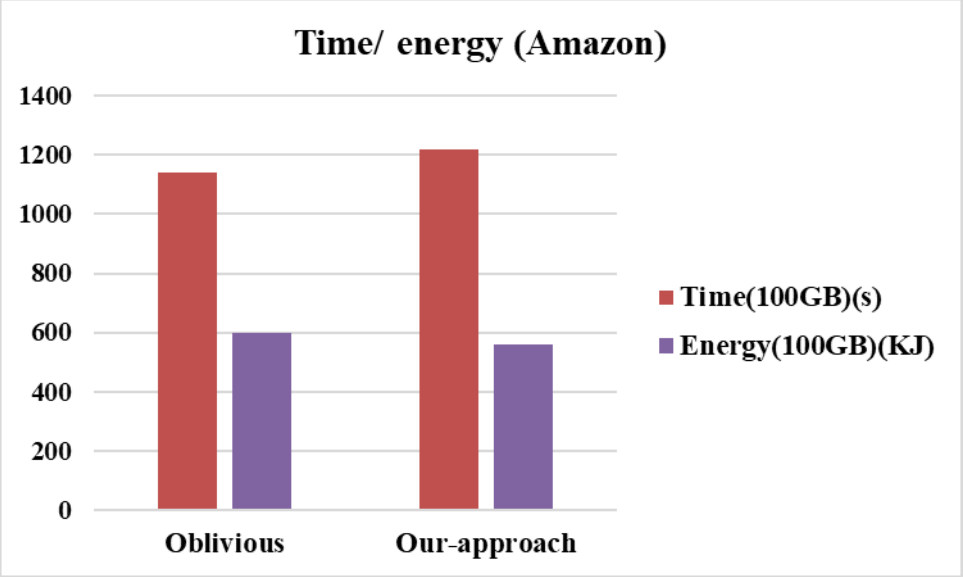

Fig. 11. Processing time and energy of Amazon datasets

Based on the results presented in Fig. 7 to Fig. 11, our approach can surpass the default scheduler and achieve 9%, 15%, 11%, 13% and 7% improvement for energy consumption in WordCount, Grep, Inverted Index, TPC and Amazon benchmarks.

Sensitivity analysis.

We also analyse the impact of data skew and the deadline on the performance of our work. For modeling data skew, we have used a mathematic law and for the deadline, we have considered two conditions.

Sensitivity to the data skew. In this case, our approach can save more energy and we have a better choice to apply the DVFS technique. Uneven distribution causes data variety/ skew among data, especially in Big Data. Aggregating data from multiple sources intensifies data variety/ skew. Fig. 12 shows the effect of aggregating data from multiple sources.

Modeling data skew. We have used Zipfian [30], [31] distribution to generate skewed data. Zipf's law states that out of a population on N elements, the frequency of elements of rank k, f(k;z,N) is:

$$f(k;z,N) = \frac{\frac{1}{k^z}}{\sum_{n=1}^{N}(\frac{1}{n^z})}$$

Following the Zipfian distribution, the frequency of occurrence of an element is inversely proportional to its rank.
In the current context, let:
1) N = total number of input partitions;
2) k be their rank; partitions are ranked as per the number of records in the partition that satisfy the given predicate;
3) z be the value of the exponent characterizing the distribution.

We have considered z=0 for uniform distribution and z=2 for high skew.

As shown in Fig. 12, in the case of existing data skew, our approach can perform better results. The horizontal axis shows the benchmarks and the vertical shows the normalized processing time and energy consumption. The processing time and energy consumption are normalized to the oblivious approach with uniform distribution.

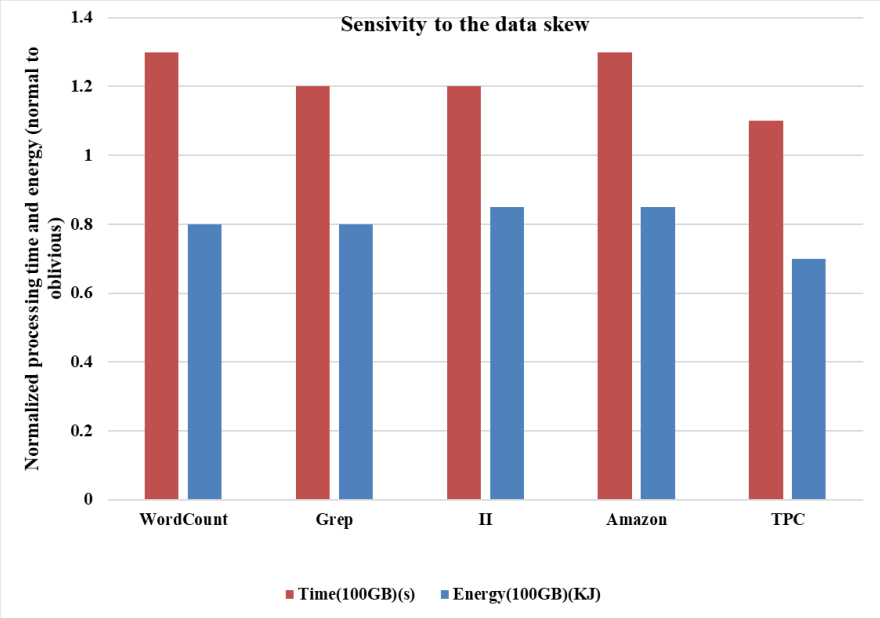

Fig. 12. Sensitivity analysis to data skew

Sensitivity to the Deadline. While there is a tight deadline, we have limited choice to apply DVFS to the computer node. Our approach has better performance in case of firm deadline.

Table 3. Tight and Firm Deadline for benchmarks

Benchmarks	Tight Deadline(s)	Firm Deadline(s)
Wordcount	1350	1500
Grep	670	730
Inverted Index	27000	30000
TPC	1250	1400
Amazon	1150	1350

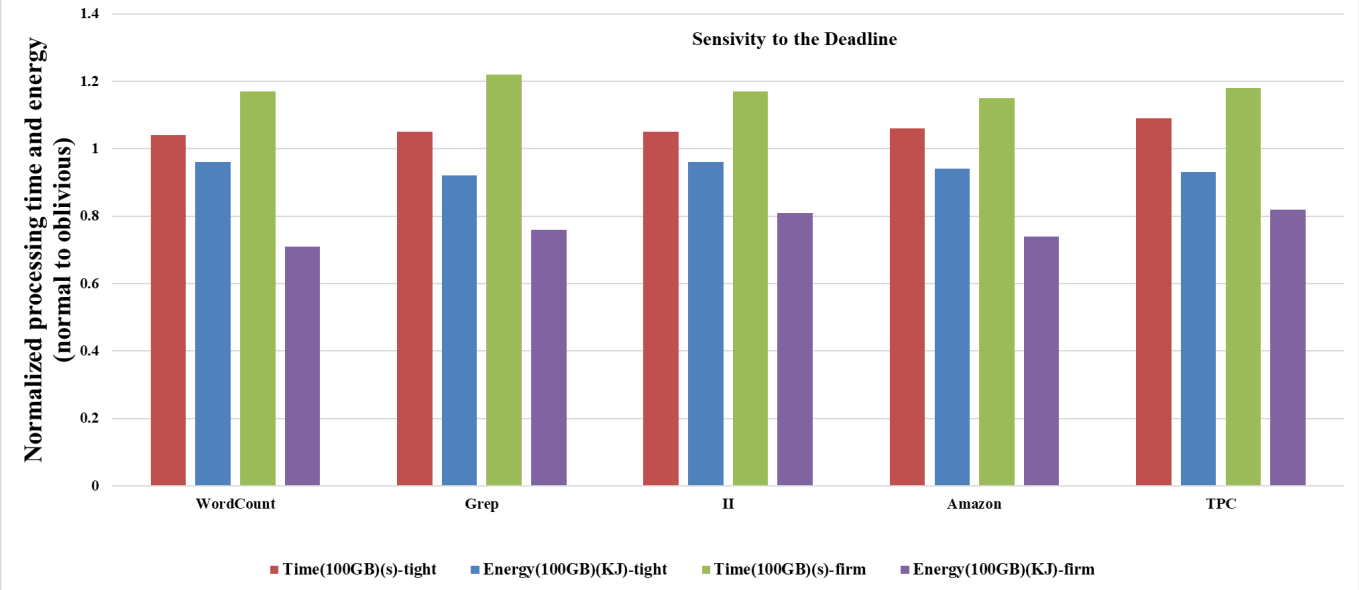

Fig. 13. Sensitivity analysis to Deadline

As Fig. 13 shows, our approach has better performance in case of a firm deadline. In this condition, we should apply DVFS technique to more parts of data in compared with the tight deadline.
At the end of this section, we should discuss about two important issues: Overhead and the Usages

- **Discuss about the overhead.** Our approach is a very low overhead solution. Sampling has less than 1% overhead for generating 5% error margin and 95% confidence interval.

- **Discuss about the usages.**
  - This approach is applicable for cloud service provider and every cloud user that can manage the infrastructure.
  - Based on the variety that is one of the features of big data, this approach could be used for processing the big data applications.
  - This approach reduces the energy consumption and the cost of energy. So, cloud providers clearly can benefit from it.
  - In this paper, we have presented an approach for reduction energy consumption in Big data processing for accumulative applications. We have presented the definition of accumulative application in [22]. This type of applications is an important type of Big Data applications [2], [22].

6 CONCLUSION

In summary, we have studied the impact of data variety on CPU utilization and energy consumption for Big Data processing. We divide input data into some same size blocks and via sampling estimate the performance of each block processing under different DVFS plan.
The results show the success of our variety- conscious approach in compared to another approach. Based on the results, in case of firm deadline, we able to apply the DVFS technique to more parts of data. So, our approach generates better results in compared with tight condition.
Many interesting directions exist to continue this work. First, considering energy price in various parts of data and geographical area. Based on this idea, we can process input data when/ where the energy price is minimum and improve the big data processing cost. Second, we can consider solar energy for our processing. So, we can process the main part of our input data by near-free energy.

Abbreviations

D: Deadline
EC: Energy Consumption
FT: Finish time
UF: Finish time
TS: Time Slot
$\mathbf{B_i}$: The i-th block
$\mathbf{PT_i}$: The processing time of i-th block
RPC: Required Power for Processing
REP: Required Energy for Processing
$\mathbf{SFB_i}$: Suitable Frequency for processing B_i

Authors' contributions

HA is the primary researcher for this study. His contributions include the original idea, literature review, implementation and initial drafting of the article. FF discussed the results with the primary author to aid writing of the evaluation and conclusion sections and played an essential role in editing the paper. MF help to improve the research concept and played a crucial role in the research. All authors read and approved the final manuscript.

Acknowledgements

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

BigDataBench: http://prof.ict.ac.cn/.

TPC Benchmark: http://www.tpc.org/information/benchmarks.asp.

Amazon product data: http://jmcauley.ucsd.edu/data/amazon/.

IMDB data files: https://datasets.imdbws.com/.

Gutenberg datasets: https://www.gutenberg.org/.

Quotes-dataset: https://www.kaggle.com/akmittal/quotes-dataset.

Funding

Not applicable.

REFERENCES

[1] "Cost of Power in Large-Scale Data Centers," 4 Nov. 2018. [Online]. Available: https://perspectives.mvdirona.com/2008/11/cost-of-power-in-large-scale-data-centers/.

[2] H. Ahmadvand and M. Goudarzi, "Using Data Variety for Efficient Progressive Big Data Processing in Warehouse-Scale Computers," *IEEE Computer Architecture Letters,* vol. 16, no. 2, pp. 166 - 169, 2017.

[3] Í. Goiri, K. Le, T. D. Nguyen, J. Guitart, J. Torres and R. Bianchini, "GreenHadoop: leveraging green energy in data-processing frameworks," in *EuroSys '12 Proceedings of the 7th ACM european conference on Computer Systems*, Bern, Switzerland, 2012.

[4] Y. Ying, R. Birke, C. Wang, L. Y. Chen and N. Gautam, "Optimizing Energy, Locality and Priority in a MapReduce Cluster," in *2015 IEEE International Conference on Autonomic Computing*, Grenoble, France, 2015.

[5] A. Verma, L. Cherkasova and R. H. Campbell, "Orchestrating an Ensemble of MapReduce Jobs for Minimizing Their Makespan," *IEEE Transactions on Dependable and Secure Computing,* vol. 10, no. 5, pp. 314 - 327, 2013.

[6] S. Maroulis, N. Zacheilas and V. Kalogeraki, "A Framework for Efficient Energy Scheduling of Spark Workloads," in *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, Atlanta, GA, USA, 2017.

[7] A. Acosta, F. Almeida and V. Blanco, "A heuristic technique to improve energy efficiency with dynamic load balancing," *The Journal of Supercomputing,* p. 1–15, 2018.

[8] K. Duan, S. Fong, W. Song, A. V. Vasilakos and R. Wong, "Energy-Aware Cluster Reconfiguration Algorithm for the Big Data Analytics Platform Spark," *Sustainability ,* vol. 12, no. 9, p. 2357, 2017.

[9] S. Ibrahim, T.-D. Phan, A. Carpen-Amarie, H.-E. Chihoub, D. Moise and G. Antoniu, "Governing Energy Consumption in Hadoop through CPU Frequency Scaling: an Analysis," *Future Generation Computer Systems,* vol. 54, pp. 219-232, 2016.

[10] M. W. Azhar, P. Stenström and V. Papaefstathiou, "SLOOP: QoS-Supervised Loop Execution to Reduce Energy," *ACM Transactions on Architecture and Code Optimization ,* vol. 14, no. 4, 2017.

[11] Q. Fettes, M. Clark, R. Bunescu, A. Karanth and A. Louri, "Dynamic Voltage and Frequency Scaling in NoCs with Supervised and Reinforcement Learning Techniques," *IEEE Transactions on Computers,* 2018.

[12] D. R. Sulaiman, M. A. Ibrahim and I. Hamarash, "DYNAMIC VOLTAGE FREQUENCY SCALING (DVFS) FOR MICROPROCESSORS POWER AND ENERGY REDUCTION," in *th International Conference on Electrical and Electronics Engineering*, 2005.

[13] P. Arroba, J. M. Moya, J. L. Ayala and R. Buyya, "Dynamic Voltage and Frequency Scaling-aware dynamic consolidation of virtual machines for energy efficient cloud data centers," *Concurrency and Computation: Practice and Experience,* vol. 29, no. 10, 2017.

[14] H. Liu, B. Liu, L. T. Yang, M. Lin, Y. Deng, K. Bilal and S. U. Khan, "Thermal-Aware and DVFS-Enabled Big Data Task Scheduling for Data Centers," *IEEE Transactions on Big Data,* vol. 4, no. 2, pp. 177 - 190, 2017.

[15] Y. Sfakianakis, C. Kozanitis, C. Kozyrakis and A. Bilas, "QuMan: Profile-based Improvement of Cluster Utilization," *ACM Transactions on Architecture and Code Optimization (TACO),* vol. 15, no. 3, 2018.

[16] G. Matheou and P. Evripidou, "Data-Driven Concurrency for High Performance Computing," *ACM Transactions on Architecture and Code Optimization ,* vol. 14, no. 4, 2017.

[17] A. N. Asadi, M. A. Azgomi and R. Entezari-Maleki, "Evaluation of the impacts of failures and resource heterogeneity on the power consumption and performance of IaaS clouds," *The Journal of Supercomputing,* p. 1–25, 2018.

[18] J. Son, A. V. Dastjerdi, R. N. Calheiros and R. Buyya, "SLA-Aware and Energy-Efficient Dynamic Overbooking in SDN-Based Cloud Data Centers," *IEEE Transactions on Sustainable Computing,* vol. 2, no. 2, pp. 76 - 89, 2017.

[19] K. Liu, G. Pinto and Y. D. Liu, "Data-Oriented Characterization of Application-Level Energy Optimization," in *International Conference on Fundamental Approaches to Software Engineering*, 2015.

[20] Z. Li, S. Tesfatsion, S. Bastani, A. Ali-Eldin, E. Elmroth, M. Kihl and R. Ranjan, "A Survey on Modeling Energy Consumption of Cloud Applications: Deconstruction, State of the Art, and Trade-off Debates," *IEEE Transactions on Sustainable Computing,* vol. 2, no. 3, pp. 255 - 274, 2017.

[21] H. Ahmadvand and M. Goudarzi, "SAIR: Significance-Aware Approach to Improve QoR of Big Data Processing in case of Budget Constraint," *The Journal of Supercomputing,* 2019.

[22] H. Ahmadvand, M. Goudarzi and F. Foroutan, "Gapprox: Using Gallup Approach for Approximation in Big Data Processing," *Journal of Big Data,* vol. 6, no. 1, 2019.

[23] L. Wang, Z. Jianfeng, L. Chunjie, Z. Yuqing, Y. Qiang, H. Yongqiang and G. e. a. Wanling, "Bigdatabench: A big data benchmark suite from internet services," in *In High Performance Computer Architecture (HPCA), 2014 IEEE 20th International Symposium on*, 2014.

[24] "TPC," [Online]. Available: http://www.tpc.org/default.asp. [Accessed 30 Sept 2018].

[25] "Amazon product data," [Online]. Available: http://jmcauley.ucsd.edu/data/amazon/. [Accessed 30 Sept 2018].

[26] "IMDb data files," [Online]. Available: https://datasets.imdbws.com/. [Accessed 30 Sept 2018].

[27] "Project Gutenberg," [Online]. Available: http://www.gutenberg.org/. [Accessed 30 Sept 2018].

[28] "quotes-dataset," [Online]. Available: https://www.kaggle.com/akmittal/quotes-dataset. [Accessed 30 Sept 2018].

[29] B. Efron and R. Tibshirani, Bootstrap methods for standard errors, confidence intervals, and other measures of statistical accuracy, Statistical science, 1986.

[30] D. E. Knuth, The Art of Computer Programming: Volume 3: Sorting and Searching, Addison-Wesley; Volume 3 edition (1973), 1973.

[31] R. Grover and M. J. Carey, "Extending Map-Reduce for Efficient Predicate-Based Sampling," in *2012 IEEE 28th International Conference on Data Engineering*, Washington, DC, USA, 2012.