RESEARCH ARTICLE OPEN ACCESS

Data Intensive Task Analysis using Dynamic Voltage and Frequency Scaling Governors

Miss. Rucha Shankar Jamale [1], Mrs. Sunita Dhotre [2], Dr. Suhas .H Patil [3]
M.Tech [1], Associate Professor [2], Professor [3]
Department of Computer Engineering
Bharati Vidyapeeth Deemed University College of Engineering
Pune - India

ABSTRACT
Data Intensive Tasks are the tasks whose primary entities are purely data oriented. All the Data Intensive Applications are user defined, and I/O bound. The word count application is considered as a data intensive application. This paper gives a brief analysis of Word Count Program using Dynamic Voltage & Frequency Scaling Governors.
Keywords:- Data Intensive Task, Word Count, Dynamic Voltage & Frequency Scaling (DVFS), Governors.

I. INTRODUCTION

Data-intensive tasks [1] are used to describe applications that are I/O bound [2] or with a need to process large volumes of data. This aspect gives the complete idea about their I/O movement, overall processing time as well as data manipulation [3] [27].
Parallel Computing [4] approach is seen in Data Intensive platforms which eventually combines multiple disk and processors into massive computing clusters connected to high-speed communication networks [5].

The Word Count [6] terms refer to the appropriate numbers in an individual document, book or thesis. Word Count is essential, especially where there are particular limitations on the text. Many tools and programs already exist to measure the actual and exact word count. The modern tools [7] excel in measuring lines and characters as well.
The Word Count Application [8] deal only the amount of text or data included in it and accordingly analyzes the result hence is purely an Data Intensive Application.

II. RELATED WORK

A. Data Intensive Tasks

The data intensive tasks use Data intensive computing mechanism. And Data Intensive Computing is carried out by using parallel computing applications, while the parallel computing applications use parallel data approach to process large amount of data typically it is also referred as Big Data.[9][28]

The application or task can be considered as Data intensive if they devote maximum processing time [10] to I/O and data movement and manipulation. The processing of data intensive application is done by Parallel processing mechanism in which the data is partitioned or subdivided into multiple data segments and are then independently processed on an appropriate computing platform [11] by executing a particular application program in parallel after that all the results are assembled to produce a complete output. Parallel processing [12] benefits more and more when large aggregate data distribution is carried out. The actual challenge beholds when large amount of data is parallel processed and simultaneously managed to thrive desired results with correct output.

The data parallelism [13] applies computation independently to each data set. The main aim of developing a Data intensive application is they possess scalable performance and can improve several levels of performance magnitude. The key confronts for developing a Data intensive application are the choice of accurate algorithm, its substantial programming complexity [14] and limitations to accomplish target architecture. Designing a Data intensive platform provides efficiency, reliability, scalability and availability.

The Data Intensive Computation clarifies following characteristics:

1) The selection of algorithm, data, and programs used for computation
2) Utilization of respective programming model in an efficient way
3) Concern regarding availability and reliability
4) The innate scalability of primary software and hardware architectures.

B. Dynamic Voltage & Frequency Scaling

The power management scheme focuses on two aspects Dynamic Power Management (DPM) and Dynamic Voltage and Frequency Scaling (DVFS) [15]. The DPM deals with executing the high workload at a maximum CPU speed while remaining workload at low power mode. The DVFS deals with executing processes at a low-performance setting regarding voltage and frequency.
DVFS techniques [16][29] are widely applied in smartphones to reduce power consumption by changing CPU core frequency and system voltage, and eventually, this results in variance in response time in smartphones while executing a precise application.

Many CPU Frequency Scaling Governors exist which allows the drives to set the target frequency. For the efficient use of CPU dynamic frequency scaling mechanism [17] is applied. These governors are embedded in patched Linux kernel System.
DVFS schemes include governors [18] like Ondemand governor, Performance governor, Conservative Governor, Powersave Governor and Interactive governor.

Ondemand governor [19] is the default governor of maximum Android-based smartphones. Ondemand governor was introduced in the Linux Kernel 2.6.10. Depending on the processor utilization it dynamically changes the processor frequency. The use of the processor is checked, and if the value exceeds the threshold, this governor set the frequency to the highest available value. If the utilization is less than the threshold, the governor steps down the frequency. The range of frequencies can be controlled by the governor and also the rate of checking the utilization of the system.

Performance governor sets the frequency to the highest frequency which is available. This allows the processor clock [20] speed to be set to maximum thus allowing maximum performance. No power savings are achieved which Performance Governor is used, but it allows changing the frequency.

In Conservative Governor Frequency is dynamically adjusted based on the processor utilization with a gradual increase in its value. The frequency of the processor utilization is checked and if its lies below or above the utilization thresholds, this governor steps up or down the frequency to the next available instead directly going to high or low.

Powersave Governor sets the processor to the lowest available frequency however a range of frequencies can be adjusted. The process runs at the slowest frequency.Therefore it takes the time to go idle.

In Userspace Governor frequency is set manually in this governor. It does not dynamically change the frequency. Compare to all other governors Userspace is more customizable, it has a most efficient way for balancing between Performance and power of the system.

C. Completely Fair Scheduler

The ultimate goal of CFS is to provide the fair amount to all the tasks proportional to their weights.
CFS is a virtual runtime scheduler. The CFS algorithm uses Red-Black Tree, in this tree the tasks are sorted in a tree form from left to right according to the increasing order of their respective virtual run times. Meanwhile, CFS executes its task initiating from left most leaf moving towards the right.
The proposed system [21] focuses on estimation of response time analysis by designing scheduler driven DVFS scheme. Response Time Analysis of Linux Kernel Completely Fair Scheduler [22][23][24] for Data Intensive Task is carried out by analysis of frequency change by the help of DVFS properties invoking in Linux kernel with the help of Data Intensive Task. To optimize the user experience the Completion time or Response time of a Process is the main focus of the work. For the given frequency limits the utility of CPU Scheduling Algorithm will be explored.

III. PROPOSED SCHEME

In the proposed scheme word count is a Data Intensive Task is explained with the help of DVFS scheme. The results are gathered by setting different DVFS governors one by one. In the proposed theory, here only two governors are used for analysis- Performance and Powersave.
Benchmarking [25] process is also computed here. Benchmarking helps to measure the performance of the system as well as compares it with other system results. Hackbench [26] benchmarking tool is used here. Hackbench is a benchmarking tool and a stress test widely used in Linux Kernel Scheduler.

The whole implementation and result analysis are done on Linux platform Ubuntu 15.10 which helped to accomplish appropriate results. Hence the proposed work deals at Kernel level which is unique and assisted in getting better output.
The implementation starts by implementing a word count program which is written in C programming language through

which an input text file of 103.4MB is invoked. The word count program gives the overall count of number of words, number of lines and number of characters. Hackbench is installed and built. The DVFS governors are set through a command line and then the graphs are plotted using time command.
Above procedure when followed gives the accurate graphical analysis of the word count program while executing it by setting different governors on single core, dual core, triple core, and quad core respectively. The average values are mentioned in the following table

Governors	User	System	Percentage
Performance	0.994	0.0092	99.4%
Powersave	0.992	0.0106	99.2%

Table.1 User and System average of Governors

The acquired results by following above procedure are as given below. Two graphs are calculated here by setting two different DVFS governors Performance and Powersave respectively.

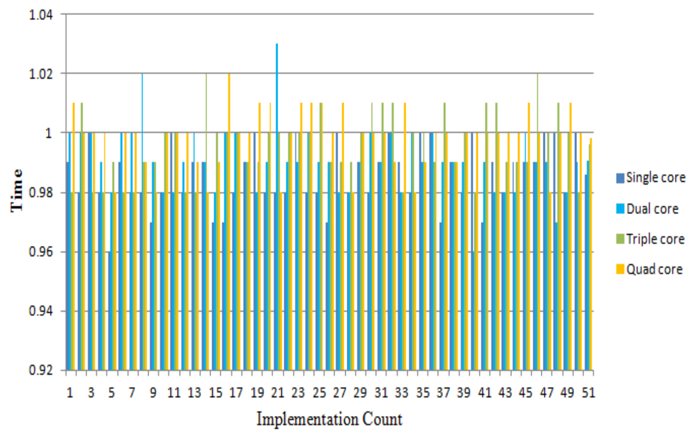

Fig.1 Word count program analysis using Performance governor

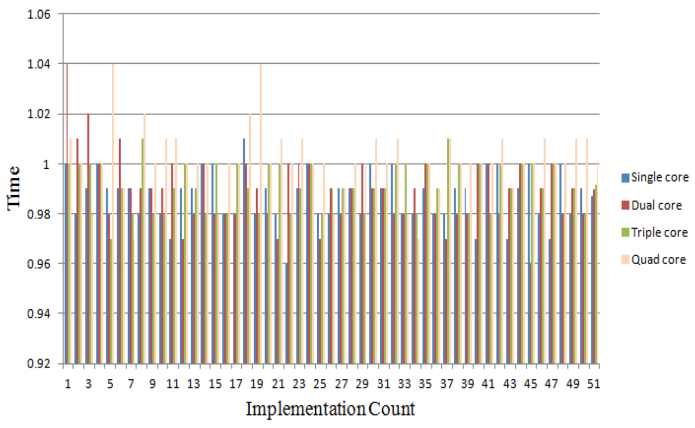

Fig.2 Word count program analysis using Powersave governor

The variation between time vs. number of implementations is shown by executing the program on different cores. Fig.1 depicts the analysis of Performance governor which eventually proves that performance governors thrives to the highest frequency in all cores compare to powersave governor. The analysis of Powersave governor is shown in fig.2 which proves that powersave governor works at lowest frequency while executing the program on different cores respectively. Hence, from Table.1 the proposed work proves that Performance governor utilization is 99.4% while Powersave governor is 99.2% and hence, performance governor frequency is greater than powersave frequency.

IV. CONCLUSION

Several Data intensive applications are implemented and used for obtaining precise results. The proposed work here does the analysis directly on Linux Kernel aspect by the help of DVFS governors. Hence a word count program is proved to be a data intensive task with respect to different Dynamic Voltage & Frequency Scaling Governors through graphical representation.
The future work for the proposed scheme can be justified by including DVFS scheme in CFS scheduling algorithm.

ACKNOWLEDGMENT

The authors would like to thank all the Bharati Vidyapeeth College of Engineering staff members for their valuable inputs and support.

REFERENCES

[1] Singh, Hartej, et al. "MorphoSys: an integrated reconfigurable system for data-parallel and

computation-intensive applications." *IEEE transactions on computers* 49.5 (2000): 465-481.

[2] Lee, Walter, et al. "Implications of I/O for gang scheduled workloads." *Job Scheduling Strategies for Parallel Processing*. Springer Berlin/Heidelberg, 1997.

[3] Menon, Jaishankar, David Pease, and Robert Rees. "Distributed storage system for data-sharing among client computers running defferent operating system types." U.S. Patent Application No. 10/323,113.

[4] Hendrickson, Bruce, and Tamara G. Kolda. "Graph partitioning models for parallel computing." *Parallel computing* 26.12 (2000): 1519-1534.

[5] Mascolo, Saverio. "Congestion control in high-speed communication networks using the Smith principle." *Automatica* 35.12 (1999): 1921-1935.

[6] Blumenstock, Joshua E. "Size matters: word count as a measure of quality on wikipedia." *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008.

[7] Bangert-Drowns, Robert L. "The word processor as an instructional tool: A meta-analysis of word processing in writing instruction." *Review of Educational research* 63.1 (1993): 69-93.

[8] Pennebaker, James W., Martha E. Francis, and Roger J. Booth. "Linguistic inquiry and word count: LIWC 2001." *Mahway: Lawrence Erlbaum Associates* 71.2001 (2001): 2001.

[9] Rucha Shankar Jamale.*"A Study On Near Data Processing"*, Volume 4, Issue VII, International Journal for Research in Applied Science and Engineering Technology (IJRASET) Page No: , ISSN : 2321-9653

[10] Daniels, Richard L., and Panagiotis Kouvelis. "Robust scheduling to hedge against processing time uncertainty in single-stage production." *Management Science* 41.2 (1995): 363-376.

[11] Baratloo, Arash, Partha Dasgupta, and Zvi M. Kedem. "Calypso: A novel software system for fault-tolerant parallel processing on distributed platforms." *High Performance Distributed Computing, 1995., Proceedings of the Fourth IEEE International Symposium on*. IEEE, 1995.

[12] Hwang, Kai, and Zhiwei Xu. *Scalable parallel computing: technology, architecture, programming*. Boston: WCB/McGraw-Hill, 1998.

[13] Bal, Henri E., and Matthew Haines. "Approaches for integrating task and data parallelism." *IEEE concurrency* 6.3 (1998): 74-84.

[14] Kitchenham, Barbara A. "Measures of programming complexity." *ICL Technical Journal* 3 (1981): 298-316.

[15] R. C. Garcia, J. M. Chung, S. W. Jo, T. Ha, and T. Kyong, "Response time performance estimation in smartphones applying dynamic voltage & frequency scaling and completely fair scheduler," *Proc. Int. Symp. Consum. Electron. ISCE*, vol. 2, no. 2, pp. 1–2, 2014.

[16] Choi, Kihwan, Ramakrishna Soma, and Massoud Pedram. "Dynamic voltage and frequency scaling based on workload decomposition." *Proceedings of the 2004 international symposium on Low power electronics and design*. ACM, 2004.

[17] Dhiman, Gaurav, and Tajana Simunic Rosing. "Dynamic voltage frequency scaling for multi-tasking systems using online learning." *Low Power Electronics and Design (ISLPED), 2007 ACM/IEEE International Symposium on*. IEEE, 2007.

[18] D.Brodowski, "CPUFreq Governors," https://www.kernel.org/doc/Documentation/cpu-freq/governors.txt, Nov. 2013.

[19] Pallipadi, Venkatesh, and Alexey Starikovskiy. "The ondemand governor." *Proceedings of the Linux Symposium*. Vol. 2. No. 00216. sn, 2006.

[20] Noble, James L., et al. "Adjusting clock frequency and voltage supplied to a processor in a computer system." U.S. Patent No. 5,760,636. 2 Jun. 1998.

[21] Rucha Shankar Jamale, Sunita Dhotre and Pooja Tanaji Patil "A Survey on Response Time Analysis using Linux Kernel Completely Fair Scheduler for Data Intensive Tasks" 5 th International Conference on Communications, Electrical, Electronics and Computer Engineering (ICEEC 2017) Paper ID : ICEEC802017

[22] Dilipkumar, Vora Shivani, M. Tech, and S. S. Dhotre. "Runtime CPU Scheduler Customization Framework for Real Time Operating System."

[23] Kabugade, Rohan R., S. S. Dhotre, and S. H. Patil. "A Modified O (1) Algorithm for Real Time Task in Operating System."

[24] Kabugade, Rohan R., S. S. Dhotre, and S. H. Patil. "A Study of Modified O (1) Algorithm for Real Time Task in Operating System." *Sinhgad Institute of Management and Computer Application NCI2TM* (2014).

[25] Pollard, Daniel A., et al. "Benchmarking tools for the alignment of functional noncoding DNA." *BMC bioinformatics* 5.1 (2004): 6.

[26] Zhang, Yanmin. "Hackbench." *U RL http://people. redhat. com/mingo/cfs-scheduler/tools/hackbench. c* (2008).

[27] Silberschatz, Abraham, et al. *Operating system concepts*. Vol. 4. Reading: Addison-wesley, 1998.

[28] Kumar, Vipin, et al. *Introduction to parallel computing: design and analysis of algorithms*. Vol. 400. Redwood City, CA: Benjamin/Cummings, 1994.

[29] Carroll, Aaron, and Gernot Heiser. "An Analysis of Power Consumption in a Smartphone." *USENIX annual technical conference*. Vol. 14. 2010.