Procedia Computer Science

Volume 29, 2014, Pages 62–72

ICCS 2014. 14th International Conference on Computational Science

An Empirical Study of Hadoop's Energy Efficiency on a HPC Cluster

Nidhi Tiwari[1,4], Santonu Sarkar[1], Umesh Bellur[2], and Maria Indrawan[3]

[1] Infosys Ltd., Bangalore, India
nidhi_tiwari@infosys.com, santonu_sarkar01@infosys.com
[2] Indian Institute of Technology Bombay, Mumbai, India
umesh@cse.iitb.ac.in
[3] Monash University, Melbourne, Australia
Maria.Indrawan@monash.edu
[4] IITB-Monash Research Academy, Mumbai, India

Abstract

Map-Reduce programming model is commonly used for efficient scientific computations, as it executes tasks in parallel and distributed manner on large data volumes. The HPC infrastructure can effectively increase the parallelism of map-reduce tasks. However such an execution will incur high energy and data transmission costs. Here we empirically study how the energy efficiency of a map-reduce job varies with increase in parallelism and network bandwidth on a HPC cluster. We also investigate the effectiveness of power-aware systems in managing the energy consumption of different types of map-reduce jobs. We comprehend that for some jobs the energy efficiency degrades at high degree of parallelism, and for some it improves at low CPU frequency. Consequently we suggest strategies for configuring the degree of parallelism, network bandwidth and power management features in a HPC cluster for energy efficient execution of map-reduce jobs.

Keywords:

1 Introduction

There has been a many-fold increase in computing capability of today's high performance computing (HPC) infrastructure. Nevertheless this speedup comes at an extremely high energy cost with these machines consuming peak power of 4500 kW, and electricity cost being $0.1 per kWh approximately [5, 14]. In order to manage this cost, it is essential to improve the power-performance efficiency by optimizing the amount of work done per unit energy. The processors with Dynamic Voltage Frequency Scaling (DVFS) capability and the Operating Systems (OS) with power management features are available in modern HPC system. As the power consumed by a system largely depends on the CPU frequency and utilization levels [2], these power aware systems operate at optimal CPU frequency to minimize their power consumption.

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2014

doi:10.1016/j.procs.2014.05.006

A HPC infrastructure comprising of multi core systems is increasingly being used to process massive amount of data using popular map-reduce programming model [20]. Hadoop is commonly used in the areas, like, computational biology, computational economics, and computational journalism, to achieve scientific breakthroughs from computational analysis of large data sets [8]. Given so many use cases, there is a need to analyze its energy efficiency on HPC infrastructure. Hadoop runs a large number of concurrent tasks on different nodes: map tasks to process different data sets, and reduce tasks to merge the intermediate data from different map tasks. Traditionally, the data is distributed across large number of commodity hardware[1], and the parallelism is increased by adding such nodes in the cluster. When Hadoop runs on a HPC cluster with many cores, each node in the cluster can run many map and/or reduce tasks using these cores. This approach can potentially minimize the data transfer cost and possibly increase the throughput. However, given the data intensive nature of map-reduce jobs, this can have different implications on the overall energy consumption in a HPC cluster. A common intuition is that as parallelism increases, the overall throughput and subsequently, the energy efficiency improves, as the job gets completed faster. However, the problem is more complex for a map-reduce job as it involves huge disk and network accesses in addition to CPU operations. If a job is CPU intensive, the throughput may increase to a certain point. But if the job has a good mix of CPU and disk operations, one may not be able to easily predict the throughput and the energy efficiency due to contention in disk, network, as well as memory and cache (Figure 2b). Therefore, a study of the change in performance and energy efficiency with change in degree of parallelism and network bandwidths in a HPC infrastructure for different types of workloads can provide interesting insights. Further, it is worth investigating how the features of a power-aware system can be used to improve the energy efficiency of map-reduce jobs in HPC clusters.

In this paper we present an empirical study of the energy-performance characteristics of Hadoop framework running on a Intel© Xeon based power-aware HPC cluster (where nodes can vary the CPU frequency to control their power consumption). As a result of this study, we determine the energy efficient configuration settings at different layers of the Hadoop map-reduce. Subsequently, we recommend strategies for configuring the studied parameters to improve the energy efficiency of different types of map-reduce jobs in HPC clusters. A few basic principles for improving the energy efficiency of map-reduce clusters have been highlighted.

The paper is organized as follows. The related work pertaining to our research is discussed in Section 2. In Section 3 we describe the cluster setup, workloads used, metrics measured and the map-reduce cluster configurations evaluated, in our empirical study. We analyze the experiment results in Section 3.6. Next, we discuss the strategies for configuring the parameters studied, in Section 4. Finally we conclude the paper with directions for future work.

2 Related Work

Power consumption of Hadoop infrastructure has been studied in the literature in recent times. We categorize these studies to three classes.

Energy Model: The authors of [1, 12, 16] have proposed different energy models to predict the energy consumption of map-reduce jobs.

Energy Efficiency: The work in [3, 11, 15, 24, 21, 17] aims to improve the energy efficiency of map-reduce clusters through job consolidation, data re-distribution and nodes reconfiguration which in turn reduce the idle period of nodes.

[1] Yahoo! uses Hadoop cluster with 38000 nodes
http://developer.yahoo.com/events/hadoopsummit2010/agenda.html

Hadoop configuration: Quite a few recent research work have studied the impact of change in various Hadoop configuration parameters, on the energy consumption of different types of map-reduce jobs. Our present work belongs to this category. In [4, 12, 16], authors studied the impact of change in the compression parameters and input data volume of a job. In [1], authors found that execution time and energy consumption both can be minimized at same time by tuning parameters they studied, like, the data replication factor and data block size. In [1, 23, 24], impact of change in the cluster configurations, like processor frequency, number of nodes and type of nodes, is studied. Based on the observations, recommendations for configuring the studied parameters are provided in [1, 12, 23], a linear regression model is proposed in [16] and a task scheduling algorithm is proposed in [24]. However, these studies have considered Hadoop running on a commodity hardware comprising of maximum 4 cores per node. These studies (except [23]) did not consider CPU frequency scaling or power management capabilities of OS like power governors. In contrast, we have evaluated a HPC cluster where each node has 80 cores. The results from commodity servers could not be applied directly to the HPC clusters, as the processing speed and power consumption of HPC machines are much higher. In HPC clusters, the relative difference between the data processing rate and communication rate (to disk and across network) is much higher than that in case of commodity clusters. This makes our study relevant in HPC scenarios. We have used CPU frequency control as well as "on-demand" power governor in our study. Unlike the previous work we have studied the impact of increasing parallelism on the overall energy efficiency by using higher parallel tasks on each node rather than increasing number of physical nodes. The study of impact of change in network bandwidth on energy efficiency of this data intensive framework is presented, which is first of its kind as per our knowledge.

3 Empirical Study

3.1 Objectives

The key objectives of our empirical study are listed below. To meet the stated objectives we conducted the experiments by varying the map and reduce slots/node, network bandwidth and CPU frequency in a HPC cluster.

- Determine the optimal degree of parallelism on a node for improving the energy efficiency of a map-reduce job in HPC clusters.
- Determine the relative benefits of increasing the network bandwidth on energy efficiency of different types of map-reduce jobs in HPC clusters.
- Determine the power policies to be used to improve the energy efficiency of different types of map-reduce jobs in HPC clusters.

3.2 Infrastructure Setup

The power aware HPC cluster used to conduct the experimental study consists of DVFS enabled machines [7], power management capable OS and power measurement software DCM. We have used four Intel© Xeon E7 4870 based machines. Each machine has 4 processors, and each processor has 20 hyper-threaded cores running at a peak frequency of 2.26 GHz. These machines are not energy proportional [2], their peak to idle power ratio is 1.7. Each machine has 128 GB of RAM and 500 GB of disk-space. They are connected via 100 Mbps and 1 Gbps LAN switches for different experiments, as commonly used in data centers. Ubuntu 12.04.2 LTS

OS is installed on all machines, as it performs DVFS based on different power/performance policies[2] using Enhanced Intel SpeedStep technology.

The cluster is equipped with a real-time monitoring toolkit called Intel Data Center Manager[3] to monitor and manage the power consumption of the cluster. The DCM console interacts with an on-board power and thermal monitoring and management device on every machine in the cluster to control the power consumed by them, based on a specific power policy. We monitor the energy consumed through the Intel DCM default web interface.

We have installed Hadoop 1.0.4 on the cluster. One machine is configured as the NameNode that also runs the JobTracker for allocating a *map* or a *reduce* task to a "slot". The rest of the machines are configured as DataNodes containing the distributed data. Each DataNode runs multiple map or reduce tasks, scheduled by the JobTracker. The logs generated by Hadoop were used to collate the job metrics like execution times and number of killed/failed tasks. The linux utilities[4] like top, iostat and vnstat, were used to monitor the system level metrics.

3.3 Workloads

In order to study the impact of selected configuration parameters on the energy efficiency of different types of jobs, we repeatedly run following popular benchmark programs -

WordCount job is mostly CPU intensive, and moderately I/O intensive in the map-phase [10]. By default it has one reduce task that collects data from all the distributed map tasks. The network traffic is not very high during map-phase but is moderately high during reduce-phase.

Sort benchmark has CPU and I/O intensive map and reduce tasks to sort and shuffle the whole data set from one order to another [10]. A high CPU, network and disk utilization is observed for a large percentage of its execution duration.

RandomTextWriter benchmark generates and writes random text on the disk in HDFS format [9]. It is used to study the energy and performance characteristics of a disk and network I/O intensive job.

3.4 Metric Used

We have measured the average completion time (CT), and the energy consumed (EC) by the application, for each scenario. To compute the performance-energy efficiency we have used the $energy$ and $delay^2$ metric, ED^2P, proposed by [18].

$ED^2P = EC \times CT^2$ metric, was originally used in the context of pipeline applications. Pradeep et al[19] recommended it for measuring energy-performance efficiency of variable voltage/frequency processors. Since a map-reduce computation is similar to a pipeline application consisting of map and reduce stages, and our cluster consists of power-aware nodes, we have adopted this metric for studying the energy efficiency of the system. The ED^2P metric is the product of energy and completion time, so lower the value of the metric, higher is the energy efficiency of the system. In rest of the paper, we call the system configurations with relatively lower value of ED^2P to be more energy efficient.

3.5 Map-Reduce Cluster Configurations

A Hadoop map-reduce deployment consists of map-reduce, Operating System (OS) and hardware layers. Each of these layers has wide range of configurable parameters. We perform

[2]Intel Enhanced SpeedStep FAQ http://www.intel.com/support/processors/sb/CS-028855.htm
[3]http://software.intel.com/sites/datacentermanager/index.php
[4]http://www.tecmint.com/command-line-tools-to-monitor-linux-performance/

experiments to determine an optimal value of one parameter at each of the layer. The parameters studied at each of the layer and the method used for examining their values are detailed below. The default settings were used for all the other Hadoop parameters.

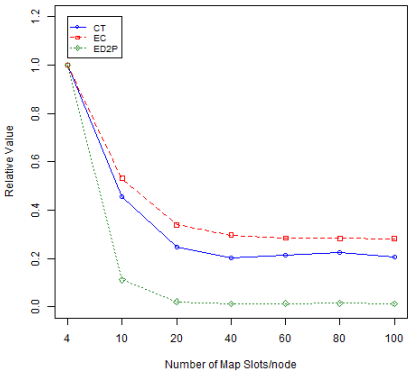
Degree of Parallelism per Node. The total number of map and reduce tasks required to finish a job, mainly, depends on the input data volume. The number of parallel slots available in the cluster then ascertain the number of iterations, also called as the number of *waves*, of parallel tasks needed to process these tasks. Therefore, higher the degree of parallelism, lesser is the number of waves and, thus, the time required to complete a job. We studied till what extent such a behavior is observed in HPC clusters by increasing the number of slots on a node. As map and reduce are different phases of a job, the results of varying map and reduce slots are analyzed separately. The number of slots per TaskTracker are increased steadily till the number of failed tasks exceeds a threshold or a node is marked unfit by Hadoop. We started the experiments with 4 slots/node for both the phases, because the nodes have four physical CPUs. We convert the actual values of CT, EC, ED^2P metrics at different levels to relative values by dividing each actual value with the value corresponding to 4 slots/node. These relative values are plotted on the Y-axis for each job in Figures 12. The default OS power governor ("onDemand") and 100 Mbps network bandwidth were used.

Network Bandwidth. A map-reduce framework processes the jobs with huge input data by distributing the data and the tasks across multiple nodes. It consolidates the distributed data in different phases based on the keys [6]. Consequently the network between the nodes becomes a critical resource for a job's performance. We performed experiments to understand the relative benefits of increasing the network bandwidth, by 900% (from 100 Mbps to 1 Gbps), on the energy efficiency of map-reduce jobs. The default OS power governor ("onDemand") and standard Hadoop configuration settings were used. The percentage improvement in CT, EC and $ED2P$ metrics from 100 Mbps to 1 Gbps are shown in Table 1 for various benchmarks.

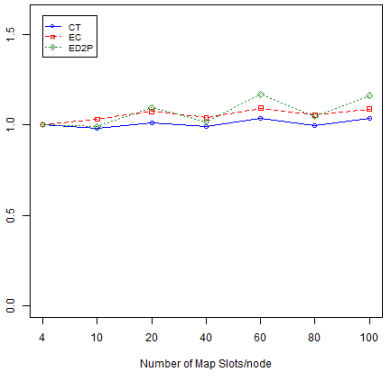
Power Management Parameters. A lower CPU frequency reduces the power consumption of a machine [19]. The default "onDemand" frequency governor in linux sets the CPU frequency based on the CPU utilization [13], so it maintains a high CPU frequency for CPU intensive processes and a low CPU frequency for others. As the CPU and I/O intensive map and reduce tasks run in parallel on a node, we compare the energy efficiency of these jobs at "onDemand" governor with that at a low frequency. The experiments were conducted by setting a constant low frequency for all the cores and "onDemand" governor, on all the nodes, in the 100 Mbps and 1 Gbps clusters. The percentage change in the metrics value from default "onDemand" governor to constant 1.06 GHz frequency, in the two clusters is shown in Figure 3.

Benchmark	**CT**	**EC**	**ED2P**
Wordcount	8.09%	13.93%	27.18%
Sort	81.00%	81.84%	99.34%
RandomTextWriter	85.07%	82.70%	99.61%

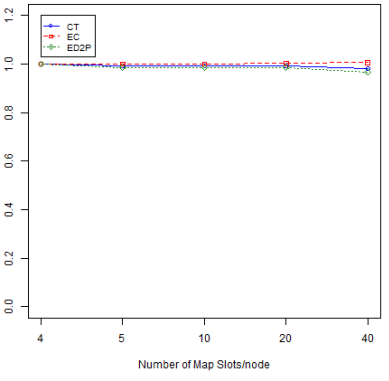
Table 1: Percentage change in CT, EC and ED^2P metrics in 1 Gbps cluster with respect to 100 Mbps cluster (higher is better)



(a) WordCount



(b) Sort



(c) RandomTextWriter

Figure 1: Relative CT, EC and ED^2P with increasing Map Slots/node (lower is better)

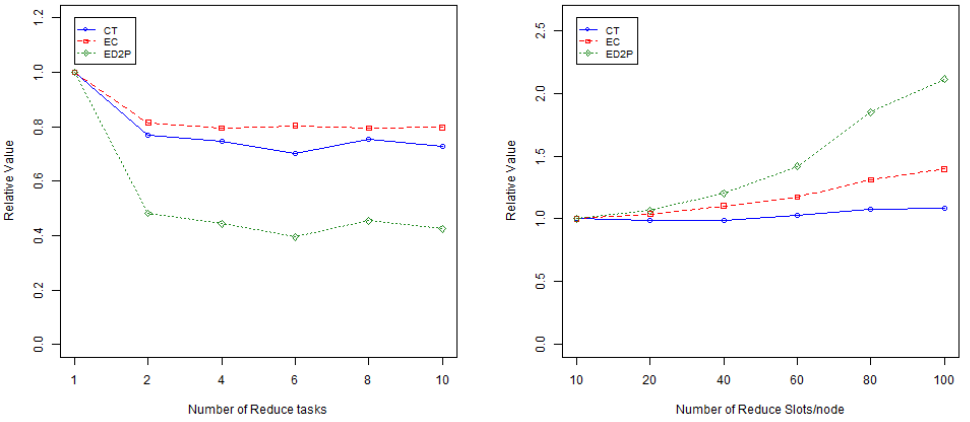
3.6 Results Analysis

3.6.1 Impact of Degree of Parallelism per Node

Varying the Map Slots/node

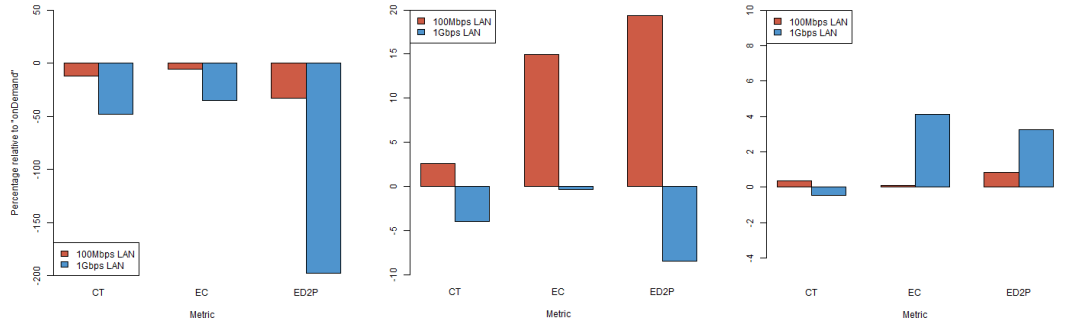
WordCount. As shown in Fig. 1a, CT, EC, ED^2P decrease rapidly till 20 map-slots/node, indicating that the increase in parallelism improves energy efficiency. As the number of map-slots are increased beyond 20, CPU and memory utilization becomes 80-90%. This leads to increase in the execution time of map tasks. So, the benefit of decrease in number of waves required to finish the job is negated; and there is only a slight reduction in CT, no change in EC and only slight improvement in energy efficiency. Beyond 40 slots/node, we observed that the number of killed map tasks, roughly, doubled, due to the increased CPU and memory contention at all the nodes. The Hadoop framework re-executed the failed tasks to finish the job successfully. As a result, the CT value increased slightly, implying that having higher number of map slots on a node does not help, though it has 80 cores.

Sort. We observe in Fig. 1b that the value of CT, EC and ED^2P hardly improve with the increase in the number of map-slots/node, even though the number of iterations required to complete all the map tasks reduce. We find that with the increase in parallelism, i) CPU



(a) Wordcount (b) Sort

Figure 2: Relative CT, EC and ED^2P with increasing Reduce slots/node (lower is better)



(a) WordCount (b) Sort (c) RandomTextWriter

Figure 3: Percentage change in CT, EC and ED^2P metrics at lower CPU frequency with respect to "onDemand" setting, in different network bandwidths (higher is better).

utilization remains 100% and disk contention increases significantly ii) network utilization becomes 100%– leading to increase in task failures and restarts. So, the time of execution, of the CPU and I/O intensive, map task increases significantly due to CPU, disk and network contentions with the increase in parallelism on each node. Consequently, CT does not reduce but the EC increases due to constant high utilization of CPU for re-execution of failed tasks.

RandomTextWriter- Map-only Job The increase in parallelism hardly had any effect on the completion time of this I/O intensive job, as shown in Fig. 1c. Because the disk access requests by multiple tasks did not get the required I/O bandwidth in a HPC machine. The total energy consumption increases by a small amount due to slight increase in the peak and average power consumption of nodes.

Varying the Reduce Slots/node

WordCount. In our cluster configuration, the maximum benefit of increasing parallelism of reduce tasks was when the number of reduce tasks were increased from 1 to 2 (Figure 2a), then

there was slight benefit till 6 reduce tasks. This was because the execution time of reduce tasks decreased initially with increase in parallelism as the data to be processed got distributed to many tasks. However, with increase in number of reduce tasks data shuffling across network increased network utilization to 95-100% at 6 reduce tasks. As consequence, no change was observed in the CT, EC and $ED2P$ metrics after 6 reduce tasks.

Sort. In case of Sort, the reduce task is both CPU and network I/O intensive as it shuffles large intermediate data from map to reduce nodes, and sorts the records again at the destination node. Consequently, the job completion time does not improve with increase in the number of reduce slots/node due to the increased CPU and network contentions. In fact the EC and $ED2P$ increase significantly with number of slots/node, as observed in Fig. 2b. This was because the number of killed and failed tasks increased at a steep rate, of almost 2-5, at high degree of parallelism due to cpu and network contentions.

3.6.2 Impact of Network Bandwidth

WordCount. We observed that the execution time of map task remained almost constant while the reduce task time improved only by 8% approximately. Therefore, the overall performance of WordCount improved only by 8% with increase in network bandwidth by 90% as it requires relatively lesser amount of data transfer across the network during execution of reduce task only. The energy consumption and efficiency improved, as shown in Table 1, due to improvement in performance and decrease in number of task failures by approximately 17%.

Sort. This benchmark is also network I/O intensive, as it transfers complete records across the network. Its response time and energy consumption reduced by approximately 81% which led to 99.34% improvement in the energy efficiency, with the 90% increase in network bandwidth, as shown in Table 1. In case of 1 Gbps network, the execution times of network intensive map and reduce tasks decreased by 22% and 90% respectively. Also the task failures, that were happening due to network contentions in 100 Mbps network, decreased by almost 60%.

RandomTextWriter. During the execution of this job, Hadoop map-reduce framework copies the whole record to multiple machines as per the replication factor. So a high network utilization of 95% was observed with 100 Mbps network. Its execution time reduced to 2.6 minutes from 17.7 minutes, with higher network bandwidth. The 85% decrease in completion time led to reduction in energy consumption and efficiency metric by almost 100%, as shown in Table 1.

3.6.3 Impact of Power Management Parameters

WordCount. Lowering the frequency of highly utilized CPU increased the completion time of WordCount as expected, increasing the amount of energy consumption and thus resulting in lesser energy efficiency, as shown in Figure 3a. However the percentage degradation was very high in case of 1 Gbps network bandwidth, almost twice of that in 100 Mbps network. This was due to a relatively high increase of 74%, in execution time of the CPU and I/O intensive reduce task, at lower CPU frequency, in case of 1 Gbps network.

Sort. On the machines with a narrow dynamic power range (of 40%), approximately 18% improvement in energy efficiency (ED^2P) of Sort was observed, when executed with low CPU frequency in 100 Mbps network, as shown in Figure 3b. The decrease in task throughput due to lower CPU frequency reduced the network contention, resulting in a significant decrease (88.88%) in number of task failures in 100 Mbps cluster. However, in case of 1 Gbps network, the network utilization was 75% on average with "onDemand" governor, so lowering the CPU frequency did not reduce the task failures significantly. Nonetheless, it increased the execution

times of map and reduce tasks by 24% and 10% respectively, and the completion time slightly, leading to a small 8.5% degradation of energy efficiency, as shown in Figure 3b.

RandomTextWriter. There was no significant improvement in energy efficiency of RandomTextWriter in case of 100 Mbps cluster, as seen in Figure 3c, as the default "onDemand" governor is able to maintain a low CPU frequency for saving power. In case of 1 Gbps network, though there is not much of impact on the completion time, small amount of energy was saved by using low CPU frequency as 100% disk and 80-90% network utilizations were observed in default mode. As a result, the energy efficiency improve by a small 3% at low CPU frequency.

4 Discussion

Our empirical study provides guidance for configuring the tested map-reduce configuration parameters to improve the energy efficiency of jobs. Here we discuss the initial set of strategies derived from our empirical study.

Configuration of Slots/node. It is commonly recommended to keep number of map and reduce slots equal to or twice of the number of cores on each node to improve resource utilization [22]. However, the analysis of the results of experiments shown in Figure 1 and 2 indicate that for energy efficiency the number of slots need not to be set equal to the number of cores on a node, specifically in a HPC cluster. In HPC cluster the data intensive map-reduce job usually become memory or I/O bound, subsequently a high number of slots/node results in task failures and re-executions. Even for a CPU intensive job, energy efficiency improves with increase in parallelism till i) the number of CPU cores are not exhausted and ii) the number of task failures remains low. Therefore, in a HPC cluster it is recommended to set:

- number of map slots/node based on the memory and disk intensity of the map tasks.
- number of reduce slots/node based on the network intensiveness of the reduce tasks.

Configuration of Network Bandwidth. The network I/O intensive map-reduce jobs get significantly higher benefit of using a high network bandwidth as compared to CPU intensive jobs in HPC clusters, as observed in Figure 1. Therefore it is recommended to use higher network bandwidth for the network I/O intensive jobs in HPC clusters for better energy efficiency. The cost of using higher network bandwidth does not provide the required benefits of improved performance and energy efficiency for CPU intensive jobs.

Configuration of Power Management Parameters. The default settings in power aware systems improve energy efficiency of distributed and parallel jobs, except a few cases. Based on a detailed study of results in Figure 3 we recommend following Power configurations for Hadoop jobs in executing HPC clusters:

- For CPU and memory intensive jobs, "onDemand" governor is an energy efficient configuration.
- For I/O (network and/or disk I/O) intensive jobs, that have less CPU utilization levels, "onDemand" governor is an energy efficient configuration.
- For I/O (network and/or disk I/O) intensive jobs, that are CPU intensive as well, a low CPU frequency improves energy efficiency.

5 Conclusion and Future Work

In this paper we have analyzed the impact of multiple configuration parameters on the energy efficiency of Hadoop map-reduce running on a power aware HPC cluster. We observed that increase in parallelism does not always translate to energy efficiency or speed-up. This is specifically true if the job is I/O intensive and the reduce phase needs a lot of data-transfer. We observed that the network bandwidth has a significant impact on the energy efficiency of the map-reduce jobs and this impact increases with the size of the intermediate data generated by the job, as expected. The study of power management features established that their deliberated use can improve the energy efficiency, of different types of map-reduce jobs. A thorough analysis of results suggests that energy efficiency of map-reduce jobs improves when the CPU processing rate is balanced against the I/O processing rate, and when minimum energy is wasted in re-executing the killed/failed tasks. We proposed the strategies to configure a HPC Cluster for different types of map-reduce jobs based on these insights.

An automated dynamic configuration manager for map-reduce clusters is in pipeline. The configuration of CPU frequency, based on the utilization levels across the cluster for the mix of jobs, will be implemented in first iteration. In order to establish further strategies to configure a map-reduce cluster for energy efficiency, we plan to perform similar empirical study for the wide range of Hadoop cluster parameters. We intend to study the impact of parallelism when the number of HPC nodes in cluster increase. We would like to conduct a comparative analysis of power-performance characteristics of a commodity cluster vis-a-vis a HPC cluster. An empirical energy consumption model will also be an important direction to investigate for determining an energy efficient configuration of Hadoop.

5.1 Acknowledgments

We would like to thank *Intel*© *India Innovation Lab* for providing the infrastructure and Subrota Mondal from HKUST, Hong Kong, for his contribution in executing a set of experiments.

References

[1] Y. Zhou B. Feng, J. Lu and N. Yang. Energy Efficiency for MapReduce Workloads: An In-depth Study. In *Proc. of Australasian Database Conference*, volume 124, pages 61–70. ACS, 2012.

[2] Luiz André Barroso and Urs Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.

[3] Yanpei Chen, Sara Alspaugh, Dhruba Borthakur, and Randy Katz. Energy efficiency for large-scale mapreduce workloads with significant interactive analysis. In *Proc. of EuroSys*, pages 43–56. ACM, 2012.

[4] Yanpei Chen, Archana Ganapathi, and Randy H. Katz. To compress or not to compress - compute vs. io tradeoffs for mapreduce energy efficiency. In *Proc. of Green networking*, pages 23—28. ACM, 2010.

[5] Wu chun Feng and K.W. Cameron. The Green500 List: Encouraging Sustainable Supercomputing. *Computer*, 40(12), 2007.

[6] Jeffrey Dean and Sanjay Ghemawat. Mapreduce: simplified data processing on large clusters. *ACM Communications*, 51(1):107–113, 2008.

[7] Rong Ge, Xizhou Feng, and Kirk W. Cameron. Improvement of power-performance efficiency for high-end computing. In *Proc. of IPDPS*. IEEE, 2005.

[8] Applications powered by Hadoop. [online]. `http://wiki.apache.org/hadoop/PoweredBy`.

[9] The Apache Hadoop Project. [online]. `http://www.hadoop.org`.

[10] Shengsheng Huang, Jie Huang, Jinquan Dai, Tao Xie, and Bo Huang. The hibench benchmark suite: Characterization of the mapreduce-based data analysis. In *Proc. of ICDEW*, pages 41–51. IEEE, 2010.

[11] Rini T. Kaushik and Milind Bhandarkar. GreenHDFS: towards an energy-conserving, storage-efficient, hybrid Hadoop compute cluster. In *Proc. of HotPower*, pages 1–9. USENIX, 2010.

[12] Willis Lang and Jignesh M. Patel. Energy management for mapreduce clusters. *VLDB Endow.*, 3(1-2):129–139, 2010.

[13] Linux Power Management. [online]. `http://doc.opensuse.org/documentation/html/openSUSE/opensuse-tuning/cha.tuning.power.html`.

[14] Kien Le, Ricardo Bianchini, Jingru Zhang, Yogesh Jaluria, Jiandong Meng, and Thu D. Nguyen. Reducing electricity cost through virtual machine placement in high performance computing clouds. In *Proc. of SC*, pages 22:1–22:12. ACM, 2011.

[15] Jacob Leverich and Christos Kozyrakis. On the energy (in)efficiency of hadoop clusters. *SIGOPS Oper. Syst. Rev.*, 44(1):61–65, 2010.

[16] Wenjun Li, Hailong Yang, Zhongzhi Luan, and Depei Qian. Energy prediction for mapreduce workloads. In *Proc. of DASC*, pages 443–448. IEEE, 2011.

[17] Nitesh Maheshwari, Radheshyam Nanduri, and Vasudeva Varma. Dynamic energy efficient data placement and cluster reconfiguration algorithm for mapreduce framework. *Future Gener. Comput. Syst.*, 28(1):119–127, 2012.

[18] Alain J. Martin. Towards an energy complexity of computation. *Inf. Process. Lett.*, 77(2-4):181–187, 2001.

[19] Modeling and Analyzing CPU Power and Performance: Metrics, Methods, and Abstractions. [online]. `http://www.princeton.edu/~mrm/tutorial/Sigmetrics2001_tutorial.pdf`.

[20] Colby Ranger, Ramanan Raghuraman, Arun Penmetsa, Gary Bradski, and Christos Kozyrakis. Evaluating mapreduce for multi-core and multiprocessor systems. In *Proc. of HPCA*, pages 13–24. IEEE, 2007.

[21] Nedeljko Vasić, Martin Barisits, Vincent Salzgeber, and Dejan Kostic. Making cluster applications energy-aware. In *Proc. of ACDC*, pages 37–42. ACM, 2009.

[22] Tom White. *Hadoop: The Definitive Guide*. O'Reilly Media, Inc., 1st edition, 2009.

[23] Thomas Wirtz and Rong Ge. Improving mapreduce energy efficiency for computation intensive workloads. In *Proc. of IGCC*, pages 1–8. IEEE, 2011.

[24] Nezih Yigitbasi, Kushal Datta, Nilesh Jain, and Theodore Willke. Energy efficient scheduling of mapreduce workloads on heterogeneous clusters. In *Proc. of GCM*, pages 1:1–1:6. ACM, 2011.