Performance and Energy Efficiency of Big Data Systems: Characterization, Implication and Improvement

Yingjie Shi
Beijing Institute of Fashion Technology
No.2 Yinghua Road, Chaoyang District, Beijing, China
shiyingjie1983@163.com

Lei Wang
Institute of Computing Technology, Chinese Academy of Sciences
No.6 Kexueyuan South Road,Haidian District Beijing,China
wl@ncic.ac.cn

Fang Du
Ningxia University
No.489 Helanshan West Road, Xixia District, Yinchuan, China
dfang@ruc.edu.cn

ABSTRACT
Large volume of data is produced by various applications in the world, processing such scale of data has great challenges in not only performance but also energy efficiency. Researchers propose various techniques to either improve the performance or the energy efficiency. The techniques of these two trends, however, are significantly different. When both performance and energy efficiency are concerned in the big data systems, how to get balance has become an issuing and challenging problem for data center administrators and hardware designers. In this paper, we conduct comprehensive evaluations on two representative platforms with different types of processors. We quantify the performance and energy efficiency, relating the evaluation results to micro-architectural activities and application characteristics. Two interesting findings are made from our evaluations: (1) the performance and energy efficiency are not only determined by the hardware technology, but also associated with the application characteristics; (2) there is no ever-victorious microprocessor in terms of both performance and energy efficiency in all the big data workloads. Based on the findings and quantified evaluation results, we provide great guidance and implications for both data center administrators and big data system designers, and we argue that a hybrid-core is an efficient way to improve the energy efficiency of big data systems with minimum performance degradation.

CCS Concepts
• Computer systems organization → Distributed architectures

Keywords
Big Data Systems; Performance; Energy Efficiency; Hybrid Core

1. INTRODUCTION
Currently, massive data is produced by various applications every day. According to the IDC white book, the total amount of data all over the world will reach 35ZB by 2020[1]. Processing such scale of data has great challenges in not only performance but also energy consumption. In a data center containing 50,000 machines, the energy and related costs can account for a third of the monthly TCO (Total Cost of Ownership)[2] and it always increases along with the increment of monthly TCO[3]. As a result, the energy cost severely limits the scalability of data centers, and it has become one of the first-class concerns in both the industry and academia.

Microprocessor manufactures pursuing various architectures and strategies toward improving the data processing performance and energy efficiency. The mechanisms to improve the performance and energy efficiency, however, are significantly different. High performance requires sophisticated techniques: faster clock speed, bigger cache size, deeper cache hierarchy, more complicated instruction execution pipeline, etc. All these techniques require more transistors and registers, which then lead to more power consumption. The core owning these techniques is called "brawny core", such as the Intel Xeon series. On the other hand, in order to reduce power consumption, different mechanisms are implemented: simpler instruction pipeline, general computation units instead of specific units, smaller cache hierarchy, etc. The core with lower power operations is always called "wimpy core", such as the Intel Atom series. Both performance and energy efficiency are concerned by users, and finding the balance between these two trends is an urging but challenging issue. There have been some research proposals for specific applications, such as web search, internet service, data mining, etc. Some proposals favor wimpy cores with the aim of high energy efficiency, and most of the applications are data-intensive. Some research work questions the efficiency of using wimpy cores and favor brawny cores, most of the work focus on the computation-intensive applications or complex data queries. The other research work proposes the hybrid of these two kinds of cores at the chip-level or system-level. In general, most of the existing work focuses on specific applications. To our best knowledge, there has been no work analyzing the performance and energy efficiency on big data applications, which owns a wide spectrum of workloads and unique characteristics.

In this paper, we conduct comprehensive evaluations of big data applications on two representative platforms with different types of processors. We first compare the performance and energy efficiency of brawny and wimpy core: Xeon E5310 and Atom D510, and relate the comparisons to micro-architectural activities to check the impacts of different processor techniques. From the results, we draw two conclusions. First, the performance and energy efficiency of big data applications are not only determined by the processor architecture or hardware technology, but also associated with the application characteristics. Second, there is no ever-victorious core in terms of both performance and energy efficiency for all the big data workloads. We argue that a hybrid-core is an efficient way to

improve the energy efficiency of big data systems with minimum performance degradation.

To explore the feasibility of hybrid core designs in big data applications, we breakdown the execution procedures of all the eight big data workloads into map and reduce phases, and analyze the characteristics of the phases in isolation. Based on the analysis on both brawny and wimpy cores, we give the trend of hybrid cores on big data workloads - assigning the map and reduce tasks to different cores according to their characteristics. We highlight the potential of the proposed hybrid core by evaluating the performance and energy efficiency of brawny core-only clusters, wimpy core-only clusters, and hybrid-core clusters. We argue that the hardware technologies are only part of the solution to improving the performance and energy efficiency of big data systems, and the workload-aware software technologies are of great importance. The main contributions of this paper include:

1) We quantify the performance and energy efficiency of hardware platforms based on brawny core Xeon E5310 and wimpy core Atom D510 with big data applications, and conduct deep analysis on how the different processor techniques and workload characteristics impact the performance and energy efficiency.

2) We break down the big data workload execution into map and reduce phase, and investigate the performance and energy efficiency characteristics of different phases on both brawny and wimpy core. Based on the analysis, we propose a novel hybrid strategy to assign map and reduce tasks to different kinds of cores.

3) We explore the feasibility and potential of the proposed hybrid strategy and present implications for the workload- aware optimization for the performance and energy efficiency of big data systems.

The rest of this paper is organized as follows. Section 2 characterizes the big data workloads used in this study. Section 3 describes our evaluation methodology. Section 4 quantifies the performance and energy efficiency of big data workloads on two representative hardware platforms, based on which we presents implications for big data system optimizations., and explores the feasibility of hybrid core designs. Related work is provided in Section 5.

2. THE BIG DTA WORKLOADS

The big data systems support comprehensive and diversified workloads, which are also changed frequently as the rapid evolution of big data applications and systems. During the evaluation of this paper, we will explore the implications of big data workloads on different big data systems, so the application type and data volume are two major dimensions during the comparison. In order to make the comparison representative and reasonable, the big data benchmark needed should meets two requirements: first, the workloads provided by the benchmark should be representative for the ongoing big data applications; secondly, the data sets should be diverse and representative in terms of both data types and sources, and the data generation tool in the benchmark should provide scalable data volumes.

In this paper, we adopt BigDataBench[4], which is an open-sourced big data benchmark suite from Internet Services[4]. BigDataBench investigates the comprehensive big data applications in the internet service field, and provide nineteen representative workloads from dimensions of both application scenarios and application types. It also provides a scalable data generation tool to create synthetical data while keeping the significant characteristics of real data. BigDataBench classifies the big data applications into three types: realtime analytics, offline analytics, and online service. We focus big data analytics in the evaluation, and choose eight workloads belonging to realtime and offline analytics, the details are shown in Table1. Three basic relational queries included select query, aggregation query and join query are chosen for realtime analytics. Five workloads are from offline analytics, they are micro benchmarks including sort, grep, and wordcount, and two basic data analytic workload in the social network and e-commerce: kmeans and naive bayes.

Table 1. Details of Chosen Workloads

Workload	Time Complexity	Characteristics
Sort	$O(n*log_2 n)$	Integer comparison and calculation
WordCount	$O(n)$	String comparison and integer calculation
Grep	$O(n)$	String comparison and integer calculation
Naïve Bayes	$O(m*n)$	Floating-point computation
Kmeans	$O(m*n)$	Floating-point computation
Select Query	$O(n)$	String comparison
Aggregation Query	$O(n)$	String comparison and integer calculation
Join Query	$O(\sum_{i=1}^{N}(n_{ri} * n_{si}))$	String comparison and integer calculation

3. EVALUATION METHODOLOGY

3.1 Experimental Platforms

During the evaluation, we choose two representative hardware platforms from brawny and wimpy fields respectively, the configuration details are shown in Table2. The Xeon E5310 belongs to brawny cores, which focuses on performance optimization and own the TDP (Thermal Design Power) of 80W. It supports out-of-order execution and four instruction issues, which are used to optimize the instruction execution speed. Also they provide relatively higher speed buses to speed up data transfer. These mechanisms require more transistors and registers, and then leads to more power consumption.

Table 2. Platform Configurations

Model	Xeon E5310	Atom D510
No. of Processors	1	1
No. of Cores /CPU	4	2
Frequency	1.6GHz	1.66GHz
L1 Cache(I/D)	32KB/32KB	32KB/24KB
L2 Cache	4096KB*2	512KB*2
L3 Cache	NONE	NONE
TDP	80W	13W
Pipeline Depth	14	16
Superscalar Widths	4	2
Architecture	X86	X86
Hyper-threading	No	Yes
Out of Order Execution	Yes	No
Specified Float Point Unit	Yes	Yes
Memory	4GB,DDR2	4GB,DDR2
Disk	SATA@7200RPM	SATA@7200RPM
Ethernet Network	1Gb	1Gb

Atom D510 is based on Intel Pine Trail architecture[5], it includes two cores with the base frequency of 1.66GHz per CPU. The Atom D510 adopts the in-order execution pipeline issuing two instructions every cycle, which requires much less transistor count to reduce power consumption. The last level cache shared by the two cores is the L2 cache of size 512KB*2. The low power operations make the Atom D510 consume much less power than the brawny cores, and its TDP is only 13W. In order to improve parallelism and make up for the lack of powerful execution architecture, it adopts the hyper-threading technology.

We construct two typical big data systems deployed Hadoop 1.0.2 based on these processors. Both of them consist of fice homogeneous nodes connected through 1Gb/s switch(one master and four slaves). In order to make the comparison fair and reasonable, and conduct the evaluation focused on CPU technologies, we follow several rules. First, we optimize the application execution performance by adjusting Hadoop configuration parameters of all the four platforms according to their hardware configurations, all the settings refer to the guidance on Hadoop official web site. Second, we collect the performance and energy efficiency results of all the slaves and normalize them by the number of CPU during the comparison. During the data processing on Hadoop, the tasks are actually executed on slaves, and master is only responsible for task scheduling with light load, which can hardly become the bottleneck in small-scale cluster. We focus on the work nodes, and don't consider the impact of master during the evaluation. CPU is the basic unit of the emerging hardware technologies, and it is also the basic processor selection unit in data center construction, so all the evaluations of this paper are normalized to the CPU level.

3.2 Metrics

The metrics required in our evaluation should be directly perceived, and what's more, they should reflect the integrated processing capacity of big data systems. We evaluate big data systems from two aspects: data processing performance and energy efficiency, and the corresponding metrics we choose are *Data-Processed-Per-Second* (*DPS* in short)[7] and *DPS-Per-Watt*. *DPS* refers to the data processed capacity in unit time, and *DPS-Per-Watt* originates from the classical metric of measuring energy efficiency called *Performance-Per-Watt*. The calculation formulas of these two metrics are shown below.

$$DPS = \frac{Data\ Input\ Size}{Run\ Time} \quad (1)$$

$$DPS - Per - Watt = \frac{DPS}{Power\ Consumption} \quad (2)$$

Data Input Size is the input data size of the workload, *Run Time* represents the execution time, and *Power Consumption* is the average power cost of running this workload. According to the definition of power, *power=energy consumed/execution time*, we can find that the computation formulas of *DPS-Per-Watt* can also be computed as the input data size divided by the energy consumed. So the *DPS-Per-Watt* can also be denoted as *DPJ*(*Data Processed Per Joule*).

3.3 Power Measurement

During the evaluation of four platforms, we compare the power efficiency from CPU level. To quantify the power dissipated by the CPU, we use an indirect power measurement method. The motherboard use specific power connector at the voltage regulator module to supply electricity for the CPU, which meets the SSI EPS 12V specification. We identify the 12V cables provide power for the CPU, and use a Hall-effect current clamp (FLUKE i30s) to measure the current of the cables, then collect the current measurement by a multimeter (FLUKE norma4000) every second. We compute the power consumed by CPU through the stable voltage and the measured current. During the preliminary exploration of CPU Hybrid, we use the power of all slave nodes to compute the power efficiency. In order to measure the power consumed by all the slave servers in the cluster, we cascade a power meter in the circuit of the servers, and collect the power measurement every second.

The processors evaluated in this paper provide hardware performance counters to measure the architectural events. We use the Linux profiling tool called Perf to collect about 20 events. In order to get the OS-level performance data, we access the proc file system and collect the data every second.

4. RESULTS AND ANALYSIS

In this section, we measure the performance and power efficiency of eight big data workloads on two typical hardware platforms, and analyze the differences of wimpy cores and brawny cores. First of all, we give the general characteristics of different workloads on all the platforms. Then we compare the Xeon E5310 and Atom D510, which are the typical brawny core and wimpy core.

4.1 The Basic Comparison of Brawny Cores and Wimpy Cores

In order to guarantee the stability of all the evaluation results, for every experiment, we run at least two times, and take the average value to eliminate bias caused by uncertainty factors. Both the performance and power efficiency metric are normalized to processor, which makes the analysis focus on the different technologies of CPU. We normalized the numbers of DPS and DPJ on each platform to that of the Atom D510, and show the ratios in Figure 1 and Figure 2.

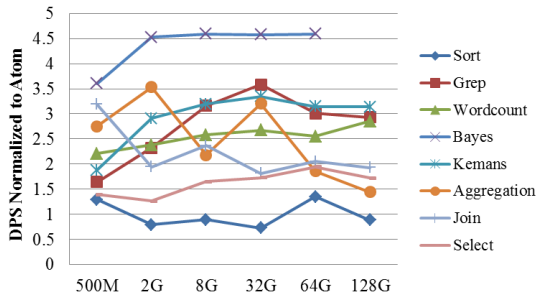

Figure 1. DPS /Processor of Each Workload Normalized to Atom

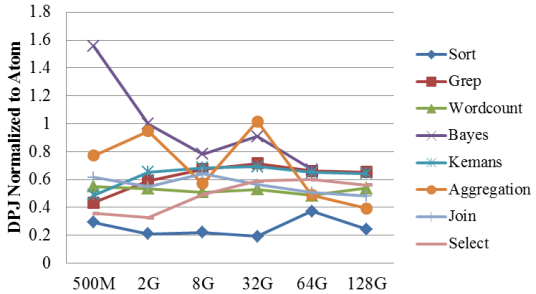

Figure 2. DPJ/Processor of Each Workload Normalized to Atom

4.1.1 *The Performance Comparison*

We set the *DPS* values of AtomD510 as the baseline, and analyze the speedup on the brawny core XeonE5310, which can be computed by *DPS of Xeon/DPS of Atom*. Generally speaking, the performance of Xeon is 1.4-4 times higher than that of Atom, and the gap between these two platforms are associated with both workloads and data size. The DPS ratios of Bayes and KMeans are the biggest, which get almost 4 times. Followed by Grep and Wordcount with 2-2.5 times higher, and the relational queries with 1.5-2 times higher. We analyze the evaluation results from two aspects: the CPU technology differences and the workload characteristics.

The basic frequency of Xeon and Atom core are 1.6 GHz and 1.66 GHz, and their hardware thread numbers on per processor are the same. The Xeon E5310 has much more sophisticated instruction pipeline than the Atom D510: Xeon E5310 issues four instructions per cycle, while the Atom D510 issues two instructions per cycle; the Xeon E5310 supports Out-Of-Order execution to speed up the instruction execution, while the AtomD510 only supports In-Order execution. Figure 3 illustrates the IPC (Instruction per Cycle) ratio of Xeon E5310 and Atom D510 of all the workloads. We can find that the IPC values of Xeon E5310 is about 3-4 times larger than Atom D510, which means that the instruction execution speed of Xeon is 4 times faster than D510. Both of these two platforms belong to X86 instruction set, and the total instruction numbers of every workload of them are similar. However, the DPS speedups of Xeon on Sort and other three relational queries are much lower than the IPC speedups. We show the CPU utilizations of all the workloads in Table 3. Generally speaking, the CPU utilization increases as data input size increases and the more stress is put on the system, and it gets stable at some data size. On the Xeon platform, the CPU utilizations of KMeans and Bayes are more than 90% at the stable point, and the CPU utilizations of Wordcount and Grep can reach 80%. However, for sort and three relational queries, the CPU utilizations are below 40% on the Xeon platform. On the Atom platform, the CPU utilizations are generally higher than Xeon, and most of them can reach 80% at the stable point, except Sort. Though the IPC values of Xeon are 4 times larger than that of Atom, the DPS speedups can not reach the same advantage for the workloads with lower CPU utilization.

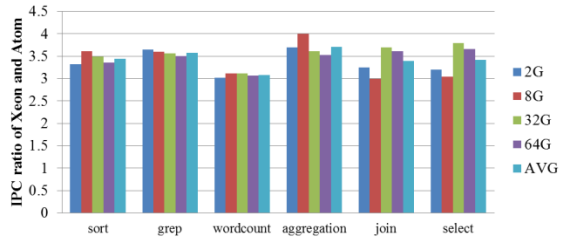

Figure 3. The Comparison of IPC between Xeon and Atom

During the evaluation experiments, we try our best to tune the Hadoop running configurations to make every workload get the optimized performance. So the next question to answer is why the CPU utilizations of some workloads on Xeon are lower? All the workloads are executed based on the MapReduce framework, and the workload execution processing include two phases: map phase and reduce phase. The input data are partitioned into disjoint splits with the same data size, which will be processed through independent and paralleled map tasks. The output of map tasks are sorted and partitioned, then transmitted to different reduce tasks. The reduce task fetches data from all the map tasks through network, and it begins the aggregation function after all the map tasks are finished. The data input size of reduce size is not fixed, it is determined by the map output ratio and the partition mechanism. We analyze the CPU utilization through break downing the workloads into map and reduce phase, the data trends are mainly determined by workload characteristics, and they are similar of different input data sizes. So we display the corresponding results with input data size 32G in Table4. We can find that on the Xeon platform, the CPU utilization of map phase of every workload is much higher than that of reduce phase, and the map phase CPU utilization gap of every workload is larger than that of reduce phase. So there are two main facts inflecting the total CPU utilization of every workload: the CPU utilization of map phase and the time percentage of map phase. For Grep, Wordcount, Naive Bayes and KMeans, the map phase time percentage is more than 90%, and their map phase CPU utilizations are more than 90%. For the other four workloads, their map phase time percentage is below 60% and even lower. What's more, compared to other workloads, the map phase computation logics of Sort, Join, and Select are simpler, which contain string comparisons without integer or float calculations. So the map phase CPU utilization of them is lower, the lower map phase CPU utilization and lower map phase time percentage make the total CPU utilization even lower. There are no reduce phase in the Select workload, which only filters the input data according to the predicate without any algebraic calculation, and its CPU utilization in the map phase is much lower than other workloads.

Table 3. CPU Utilization

		500MB	2GB	8GB	32GB	64GB	128GB
Sort	Xeon E5310	0.23	0.26	0.20	0.25	0.19	0.21
	Atom D510	0.43	0.66	0.50	0.50	0.32	0.33
Grep	Xeon E5310	0.15	0.87	0.58	0.72	0.83	0.82
	Atom D510	0.40	0.53	0.60	0.69	0.93	0
Wordcount	Xeon E5310	0.31	0.60	0.79	0.87	0.90	0.92
	Atom D510	0.60	0.83	0.93	0.94	0.96	0
Navie Bayers	Xeon E5310	0.48	0.95	0.97	0.98	0.99	0.99
	Atom D510	0.30	0.63	0.97	0.98	0.99	0
KMeans	Xeon E5310	0.30	0.65	0.83	0.92	0.92	0.95
	Atom D510	0.48	0.86	0.93	0.97	0.97	0
Aggregation	Xeon E5310	0.32	0.35	0.38	0.36	0.38	0.30
	Atom D510	0.59	0.66	0.74	0.73	0.72	0
Join	Xeon E5310	0.13	0.15	0.28	0.25	0.50	0.48
	Atom D510	0.17	0.20	0.43	0.44	0.78	0
Select	Xeon E5310	0.06	0.21	0.34	0.40	0.43	0.40
	Atom D510	0.16	0.39	0.78	0.83	0.84	0

From the above analysis, we can conclude that the Xeon E5310 shows performance advantage compared to the Atom D510, and the advantage gap are influenced by the workload characteristics: the calculation logic and time percentage of map and reduce phase. The workloads with simpler calculation and with bigger reduce phase time percentage can get less performance advantage on the brawny core.

4.1.2 *The Power Efficiency Comparison*

During our evaluation, the consumed power of Xeon E5310 is from 20W to 50W, and the consumed power of Atom D510 is from 3W to 8W. Generally speaking, the DPJ of Atom is higher

Table 4. Workload Phase Breakdown

		Map Phase				Reduce Phase			
		Time Proportion	CPU Utilization	DPS	Perf/Watt	Time Proportion	CPU Utilization	DPS	Perf/Watt
Sort	Xeon E5310	0.13	0.54	17.7	0.27	0.87	0.12	2.9	0.29
	Atom D510	0.23	0.91	8.7	0.88	0.77	0.30	2.6	0.57
Grep	Xeon E5310	0.80	0.72	16.7	0.45	0.20	0.30	8.1E-7	3.2E-8
	Atom D510	0.98	0.96	6.7	0.76	0.02	0.59	3.9E-7	5.1E-8
Wordcount	Xeon E5310	0.95	0.94	6.4	0.16	0.05	0.52	5.6	0.19
	Atom D510	0.94	0.98	2.2	0.30	0.06	0.57	2.03	0.32
Navie Bayers	Xeon E5310	0.99	0.98	0.27	0.007	0.01	0.45	0.01	0.0008
	Atom D510	0.99	0.97	0.06	0.007	0.01	0.46	0.007	0.0009
KMeans	Xeon E5310	0.98	0.96	4.9	0.22	0.02	0.38	0.0046	0.0002
	Atom D510	0.99	0.98	1.4	0.17	0.01	0.73	0.003	0.0004
Aggregation	Xeon E5310	0.42	0.85	16.6	0.45	0.58	0.39	1.73	0.06
	Atom D510	0.55	0.95	3.5	0.47	0.45	0.61	1.2	0.21
Join	Xeon E5310	0.61	0.62	9.6	0.29	0.39	0.49	8.5	0.29
	Atom D510	0.71	0.94	4.2	0.46	0.29	0.64	4.6	0.57
Select	Xeon E5310	1	0.51	16.6	0.51	0	0	0	0
	Atom D510	1	0.92	6.81	0.70	0	0	0	0

than that of Xeon, and its power efficiency advantage differs in different workloads. For the eight workloads except Naive Bayes and KMeans, Atom presents 2-5 times more power efficient than Xeon. However, in the Naive Bayes and KMeans, the Atom advantage is not obvious. The average power of Xeon is about 4-5 times higher than Atom, however, the power advantage of Atom is diminished by the performance gaps on these two workloads. It implies that though the average power of Xeon on Naive Bayes and Kmeans is higher, it cost much less time, so the total energy is similar to Atom.

From the comparison analysis of typical brawny core and wimpy core, we can draw two conclusions. First, the performance and energy efficiency of big data applications are not only determined by the processor architecture or hardware technology, but also associated with the application characteristics. Second, there is no ever-victorious core in terms of both performance and energy efficiency for all the big data workloads. We argue that a hybrid-core is an efficient way to improve the energy efficiency of big data systems with minimum performance degradation.

4.2 Phase Breakdown Analysis

Under the MapReduce framework, the workload execution processing include two phases: map phase and reduce phase. The input data are partitioned into disjoint splits with the same data size, which will be processed through independent and paralleled map tasks. The output of map tasks are sorted and partitioned, then transmitted to different reduce tasks. The reduce task fetches data from all the map tasks through network, and it begins the aggregation function after all the map tasks are finished. The data input size of reduce size is not fixed, it is determined by the map output ratio and the partition mechanism. We have shown the CPU utilization of map and reduce phase in Section 4.1, and in this subsection, we analyze the different characteristics of these two phases on all eight workloads. One thing to point out is that, during the former evaluations, we allow the reduce shuffle phase to begin before all the map tasks finish to optimize the performance. In this section, we forbid the shuffle phase to begin in advance in order to make the breakdown analysis more clear.

From the results of Table4, two observations can be gotten. First, the CPU utilization of map phase is bigger than that of reduce phase for all the workloads on both Xeon E5310 and Atom D510. This gap is more obvious in the brawny core Xeon E5310. Secondly, for most of the workloads, the DPS of map phase is bigger than that of reduce phase on both two platforms. Thirdly, the DPS gap between Xeon E5310 and Atom D510 of the map phase is bigger that of reduce phase, and the DPJ gap is just opposite. Fourthly, given the hardware platform, the differences of both the CPU utilization and DPS of map phase on different workloads are bigger than reduce phase.

We explain the observations from both the operation characteristics and instruction construction. The data processing characteristics of map phase and reduce phase are different from both data size and processing logics. The input data size of map phase is always greater or equal to that of reduce phase. For most big data analysis except sort, the map phase always filer the raw input data according to some predicate and the combine function of map phase further reduce the data transferred to reduce phase. The less input data size reduces the processing stress of reduce phase, which makes the CPU resource not fully utilized, especially in the brawny core. The data processing logic of map phase is more various and complex than the reduce phase. The input data of map phase is always the raw data, on which the map function can do more things, such as filer, sort, statistic computing, etc. While the data transferred to reduce phase is processed (key, value_list) pairs sorted by key, on which the main operation is always merge, combine, or cross product like join does. In order to further analyze the characteristics of the two phases, we investigate the widely accepted architectural metric MIPS (Million Instructions Per Second) of all the workloads, and show the results on Xeon 5310 in Figure 4. The MIPS of map phase is 2 -9 times bigger than that of reduce, which implies that the instruction behaviors of map phase and reduce phase are very different. In general, the brawny core shows more performance advantage in the map phase, while in the reduce phase, its performance advantage is not so obvious. We argue that assigning the map and reduce tasks to different cores according to their characteristics will improve the power efficiency with minimum performance degradation using present technology.

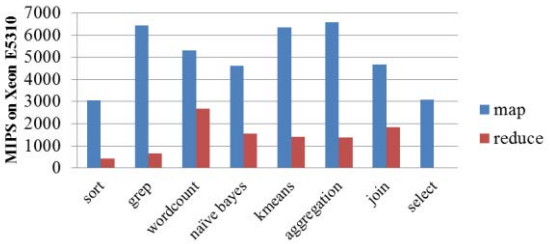

Figure 4. MIPS of Different Workloads on Xeon E5310

4.3 Preliminary Exploration of CPU Hybrid

The hybrid mechanism can be implemented through different levels: system level, processor level and core level. Considering the data network transfer cost, assigning map task and reduce task to heterogeneous core requires processor-level or core-level hybrid. Both the hybrid mechanism and task scheduling need sophisticated techniques and models, which requires further research from architecture and database management. In this paper, we quantify the performance and power efficiency on hybrid system to conduct preliminary exploration.

We have no existing brawny core and wimpy core hybrid platform on the CPU or core level, so we construct a hybrid platform as shown in Figure 5. The four slaves include two brawny-core nodes and two wimpy-core nodes. Brawny-core slaves are set to run map tasks, and wimpy-core slaves are set to run reduce tasks. This configuration will increase the data transfer cost between map and reduce tasks. In order to make the performance and efficiency comparison rational, we also make the map task and reduce task run on separated slaves on the Xeon E5310 cluster and Atom D510 cluster.

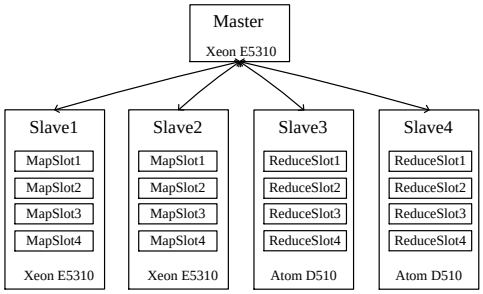

Figure 5. Hybrid Cluster Platform

The hybrid cluster consists of heterogeneous processors, so during the comparison we adopt the performance and power efficiency of the whole cluster. Because of the page limit, we choose four representative workloads to show, which include Sort, Wordcount, Grep and Bayes on 32GB data. All the results are normalized to the Atom D510, illustrated in Figure 6 and Figure7. From the results, we can find that the hybrid mechanism can improve the power efficiency with less performance cost, however, the improve effects are different for different workloads. In general, we explore how the hybrid mechanism improve the power efficiency preliminarily, and give implications for the research work on the hybrid technique. How to realize the hybrid through different levels according to the workload characteristics requires more research work, which is also our future work.

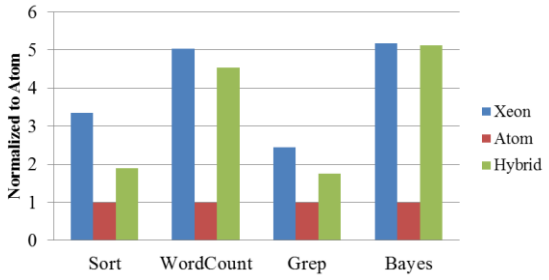

Figure 6. Energy Efficiency Comparison of Three Platforms

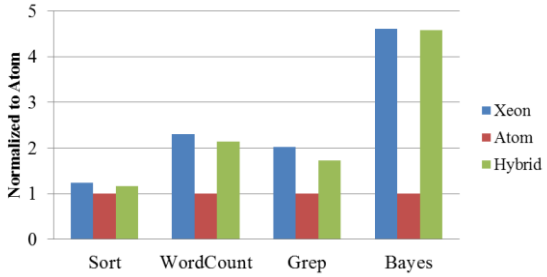

Figure 7. Performance Comparison of Three Platforms

5. RELATED WORK

As the proportion of energy cost in TCO remains increasing in data centers, more and more work is conducted on energy efficiency, and the arguments of existing work on wimpy or brawny cores can be categorized into three types. The first trend is to construct data centers based on low-powered wimpy cores with the aim of improving the energy efficiency of data processing[8-11]. In [8], the authors propose a cluster architecture called FAWN, which couples low-power, efficient embedded CPUs with flash storage to provide fast and cost-effective data access in key-value stores. Vijay et al. demonstrate the energy efficiency of wimpy cores through evaluating web search on both server and mobile-class architectures, and they also quantify the price of efficiency of wimpy cores in terms of performance and quality-of-service[9]. Compared to construct the data centers with brawny cores and improve efficiency, they favor small, low power cores and propose some system-level and microarchitectural strategies to enhance their performance. The efficiency of wimpy servers in internet-scale services is evaluated in [10], and the authors make the point that low cost, low power servers can produce the same throughput as purpose-built servers at lower energy cost. In the field of DBMS, wimpy servers are also recommended to construct energy-proportional clusters, which aims to make the consumed energy proportional to the workloads, rather than consuming large fraction of peak consumption when idle[11].

In contrast to the support voice for wimpy cores, there is also some work questioning the efficiency of using small cores in data centers [3,12,13]. In [12], the author analyzes the negative effect of switching to wimpy cores from additional software development cost, the suboptimal process scheduling and resource sharing of parallel subtasks, and the scale out limits of non-CPU cost. Willis et al. evaluate the performance scaleup of scaling out wimpy cores, and argue that the wimpy nodes exhibit disproportionate scaleup characteristics for computationally complex data query workloads[3]. The similar conclusions are reached in [13], which focuses on the analytic

workloads in Hadoop with job size less than 100GB. In general, the work holding negative attitude towards wimpy cores focuses on the computation-intensive workloads and places emphasis on the challenges and additional effort of scaling out wimpy cores.

The design of mixing two kinds of cores from different levels is also proposed. In the chip-level, the heterogeneous asymmetric multicore processors (AMP) is proposed by architects, which integrates wimpy cores and brawny cores in one processor[14-16]. In the server-level, [17] explores the performance and feasibility of the hybrid solution of platforms based on different cores. The hybrid solutions bring challenges and opportunities to the softwares, which have to be aware of the heterogeneity and fully exploit the potential of hybrid systems.

Generally speaking, the workload analysis on different cores of existing work focuses on specific applications. To our best knowledge, there has been no work focusing on the performance and energy efficiency analysis on big data workload, which contains a wide spectrum of applications and unique characteristics. What's more, the evaluations and analyses of existing work is used to support the specific arguments or proposed solutions, they focus on different aspects of application level: performance, energy efficiency, price, etc. In this paper, we conduct quantitative comparisons of different processors in the domain of big data, which covers not only application-level analysis, but also architecture- and operating system-analysis. The evaluation results and implications of this paper will bring guidance for data center provisioning and design of both architecture and software, no matter wimpy cores, brawny cores, or hybrid solutions are favored.

6. ACKNOWLEDGMENTS

This research was supported by the grants from the National Natural Science Foundation of China (No.61502279, 61363018), the Natural Science Foundation of Shandong Province (No. ZR2015FM013), the General Program of Science and Technology Development Project of Beijing Municipal Education Commission (No.KM201510012005), and the Foundation of Beijing Institute of Fashion Technology (No.NHFZ2016012). We also want to thank Jie Lv and Jing Quan from Chinese Academy of Sciences, who gave much help in the experiments.

7. REFERENCES

[1] G. J. and R. D., "Extracting value from chaos technical report white paper," Commun. ACM, 2011.

[2] Available:http://perspectives.mvdirona.com/2010/09/18/OverallDataCenterCosts.aspx

[3] W. Lang, J. M. Patel, and S. Shankar, "Wimpy node clusters: what about non-wimpy workloads?" in Proceedings of the Sixth International Workshop on Data Management on New Hardware, DaMoN 2010, Indianapolis, IN, USA, June 7, 2010, 2010, pp. 47–55.

[4] Lei Wang, Jianfeng Zhan, et al. "Bigdatabench: a big data benchmark suite from internet services," in The 20th IEEE International Symposium On High Performance Computer Architecture (HPCA-2014), February 15-19, 2014, Orlando, Florida, USA. IEEE, 2014.

[5] Available:http://www.intel.com/content/www/us/en/desktops/desktopboard- d510mo-atom-d510-brief.html?wapkw=d510, and http://www.legitreviews.com/intel

[6] Farhat et al. "Stochastic modeling and optimization of stragglers." IEEE Transactions on Cloud Computing.

[7] C. Luo, J. Zhan, Z. Jia, et al, "Cloudrank-d: benchmarking and ranking cloud computing systems for data processing applications," Frontiers of Computer Science, vol. 6, no. 4, pp. 347–362, 2012.

[8] D. G. Andersen, J. Franklin, M. Kaminsky, A. Phanishayee, L. Tan, and V. Vasudevan, "Fawn: a fast array of wimpy nodes," Commun. ACM, vol. 54, no. 7, pp. 101–109, 2011.

[9] V. J. Reddi, B. C. Lee, T. M. Chilimbi, and K. Vaid, "Web search using mobile cores: quantifying and mitigating the price of efficiency," in 37th International Symposium on Computer Architecture (ISCA 2010), June 19-23, 2010, Saint-Malo, France, 2010, pp. 314–325.

[10] J. R. Hamilton, "Internet-scale data center power efficiency," in CIDR 2009, Fourth Biennial Conference on Innovative Data Systems Research, Asilomar, CA, USA, January 4-7, 2009, Online Proceedings, 2009.

[11] D. Schall and V. Hudlet, "Wattdb: an energy-proportional cluster of wimpy nodes," in Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011, 2011, pp. 1229–1232.

[12] U. Holzle, "Brawny cores still beat wimpy cores, most of the time," IEEE Micro, vol. 30, no. 4, 2010.

[13] R. Appuswamy, C. Gkantsidis, D. Narayanan, O. Hodson, and A. Rowstron, "Scale-up vs scale-out for hadoop: Time to rethink?" in the 4th ACM Symposium on Cloud Computing, 2013.

[14] R. Kumar, K. I. Farkas, N. P. Jouppi, P. Ranganathan, and D. M. Tullsen, "Single-isa heterogeneous multi-core architectures: The potential for processor power reduction," in Proceedings of the 36th Annual International Symposium on Microarchitecture, San Diego, CA, USA, December 3-5, 2003, 2003, pp. 81–92.

[15] J. C. Mogul, J. Mudigonda, N. L. Binkert, P. Ranganathan, and V. Talwar, "Using asymmetric single-isa cmps to save energy on operating systems," IEEE Micro, vol. 28, no. 3, pp. 26–41, 2008.

[16] A. Fedorova, J. C. Saez, D. Shelepov, and M. Prieto, "Maximizing power efficiency with asymmetric multicore systems," Commun. ACM, vol. 52, no. 12, pp. 48–57, 2009.

[17] B.-G. Chun, G. Iannaccone, G. Iannaccone, R. H. Katz, G. Lee, and L. Niccolini, "An energy case for hybrid datacenters," Operating Systems Review, vol. 44, no. 1, pp. 76–80, 2010.