2017 46th International Conference on Parallel Processing Workshops

Thermal-aware Job Scheduling of MapReduce Applications on High Performance Clusters

Shubbhi Taneja, *Student Member, IEEE*, Yi Zhou, *Student Member, IEEE*,
Mohammed I. Alghamdi, *Senior Member, IEEE*, Xiao Qin, *Senior Member, IEEE*

Abstract—In this study, we develop a thermal-aware job scheduling strategy called *tDispatch* tailored for MapReduce applications running on Hadoop clusters. The scheduling idea of *tDispatch* is motivated by a profiling study of CPU-intensive and I/O-intensive jobs from the perspective of thermal efficiency. More specifically, we investigate the thermal behaviors of these two types of jobs running on a Hadoop cluster by stress testing data nodes through extensive experiments. We show that CPU-intensive and I/O-intensive jobs exhibit various thermal and performance impacts on multicore processors and hard drives of Hadoop cluster nodes. After we quantify the thermal behaviors of Hadoop jobs on the master and data nodes of a cluster, we propose our scheduler to alternatively dispatch CPU-intensive and I/O-intensive jobs. We apply our strategy to several MapReduce applications with different resource consumption profiles. Our experimental results show that *tDispatch* is conducive of creating opportunities to cool down multicore processors and disks in Hadoop clusters deployed in modern data centers. Our findings can be applied in other thermal-efficient job schedulers that are aware of thermal behaviors of CPU-intensive and I/O-intensive applications submitted to Hadoop clusters.

Index Terms—Hadoop, MapReduce, thermal profiling, thermal-aware job scheduler, data centers, benchmarking, CPU-intensive and I/O-intensive jobs, HiBench, task placement, multicore architecture

1 Introduction

Current literature focuses on the thermal efficiency of data centers [10][21], but only a handful of studies have investigated the impacts of various Hadoop jobs on the overall heat buildup in a data center. In this study, we propose a thermal-aware scheduler called *tDispatch* to improve energy efficiency of Hadoop clusters by lowering cooling cost. We show that CPU-intensive and I/O-intensive jobs exhibit distinct thermal and performance impacts on multicore processors and hard drives of Hadoop clusters. We also demonstrate that one way to make Hadoop clusters thermal friendly is to alternatively dispatch CPU-intensive and I/O-intensive jobs to Hadoop clusters; finding the most appropriate node for a particular type of a job.

Our *tDispatch* job scheduler facilitates an essential aid to data centers as *tDispatch* ensures stable building plans and energy efficiency factors. The findings of this study help in designing cost-effective cooling management solutions to modern data centers.

1.1 Motivations

There are four main motivations for working toward thermal-aware job scheduling in Hadoop clusters:

1) ability of Hadoop to manage big data.
2) augmenting cooling and electricity costs of data centers.
3) thermal profiles of Hadoop jobs running on clusters.
4) lack of thermal-aware schedulers tailored for both CPU-intensive and I/O-intensive jobs.

Modern day companies such as Amazon, Facebook, and Google regularly gather unprecedented amounts of data, often referred to as Big Data. As these companies expand their data centers, Hadoop increasingly becomes a popular big-data processing framework running in a distributed fashion on large clusters of commodity hardware. Essentially, Hadoop accomplishes two goals, namely, massive data storage and faster processing [6]. Hadoop works well with both structured and unstructured data. Many Hadoop clusters consist of enormous data sets that cannot be digitally acquired [6]. Hadoop supports MapReduce programs that spread over clusters of computers to expedite the processing of big data. This parallel computing ability, moving computation to data rather than moving data to computation, gives cost-effective Hadoop clusters an edge over the other platforms. The popularity of Hadoop motivates us to develop a thermal-aware scheduler to improve energy efficiency of Hadoop clusters.

Data centers house a large number of computer systems and components such as power supplies, network cables, thermostats, security devices, and many more. Consequently, these components generate substantial amounts of heat and consume enormous amounts of energy for cooling purposes. According to National Renewable Energy Laboratory (NREL), a data center consumes 10-100 times more energy than a regular office [7]. Two factors contributing to power usage in data centers are new generation technologies and an inclination towards multicore architectures. To increase a processor's speed, manufacturers are increasing the number of cores on each CPU chip. With a large number of cores in computing clusters, power consumption and heat dissipation have become a big concern for data centers. Also, the next generation architectures demand high power usage, which in turn translates into heat from processors. At the same time, evidence shows that the thermal-aware architecture techniques have a direct correlation with power-aware techniques [5].

Cooling of data centers is indispensable because of the

- *S. Taneja* and Y. Zhou are doctoral students at Department of Computer Science and Software Engineering in Auburn University, Auburn, Alabama 36849-5347. E-Mail: yzz0074@auburn.edu *corresponding author: shubbhi@auburn.edu*
- *X. Qin is with the Department of Computer Science and Software Engineering, Shelby Center for Engineering Technology, Samuel Ginn College of Engineering, Auburn University, AL 36849-5347. E-mail: xqin@auburn.edu.*
- *M. Alghamdi is with the Department of Computer Science, Al-Baha University, Al-Baha City, Kingdom of Saudi Arabia. E-mail: mialmushilah@bu.edu.sa.*

1530-2016/17 $31.00 © 2017 IEEE
DOI 10.1109/ICPPW.2017.46

IEEE computer society

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.

aforementioned reasons. Importantly, thermal management improves lifetime of hardware deployed in data centers. Variations in temperature accounts for more than 50% of electronic failures [1]. A recent study discovered that when a data center's temperature is over 21°C, every 10°C rise in temperature reduces hardware reliability by 50% [13]. Temperature is in direct co-relation with power output over time and; therefore, thermal management techniques are of importance in designing data centers [5].

Many existing thermal-aware schedulers were built at the processor level. Unfortunately, thermal-aware schedulers tailored for both CPU-intensive and I/O-intensive jobs are still in their infancy; no prior job scheduler exists for these two types of jobs running on clusters. This open problem motivates us to design *tDispatch* that schedules CPU-intensive and I/O-intensive jobs submitted to a dispatch queue of a cluster.

Big data applications run on large clusters deployed in data centers. Hadoop, an open-source implementation of MapReduce, is a popular choice for processing big data . Although Hadoop's Fair Scheduler assigns a fair share of CPU resources to all the jobs with minimum starvation, the existing scheduler does not take thermal efficiency into consideration. To address this critical concern for data centers, we propose *tDispatch* to cool down CPU and disk temperatures of nodes in Hadoop clusters by dispatching jobs in a thermal-friendly manner.

1.2 Contributions

One salient feature of our *tDispatch* scheduler is that it can be easily incorporated and deployed in the data centers housing Hadoop clusters. We apply *tDispatch* to schedule a group of Hadoop jobs exhibiting different resource consumption profiles. *tDispatch* judiciously reduces the temperatures of multi-core processors and disks of data nodes by altering the order in which CPU-intensive and I/O-intensive jobs are executed in a batch. We conduct extensive experiments to analyze the thermal efficiency and performance of the proposed *tDispatch* on a real-world Hadoop cluster. The experimental results confirm that *tDispatch* improves the overall thermal efficiency of the data nodes in the tested Hadoop cluster.

Our research is important in the area of thermal management as we propose a thermal-aware job scheduling strategy to dispatch both CPU-intensive and I/O-intensive jobs to a Hadoop cluster. The contributions of this study are summarized below:

- We study the thermal behaviors of MapReduce applications running on Hadoop clusters.
- We design *tDispatch* - a thermal-efficient job scheduler to dispatch CPU-intensive and I/O-intensive jobs to Hadoop clusters.
- We conduct experiments and compare the thermal efficiency and performance of *tDispatch* with those of the other two scheduler counterparts, namely, *CPUF* and *IOF*.

1.3 Roadmap

The remainder of the paper is organized as follows. Section 2 presents prior studies that are related to our *tDispatch* scheduler. Section 3 describes the experimental setups followed by the analysis of the thermal behaviors of Hadoop applications. The description of the *tDispatch* scheduler can be found in Section 4. Section 5 shows the comparisons of *tDispatch* and the two job scheduler counterparts. We discuss future research directions in Section 6. Finally, Section 7 summarizes our results and concludes the paper.

2 Related Work

2.1 Thermal-Efficient and Energy-Efficient Computing

Recently, much attention has been paid to monitoring high temperatures inside data centers. One of the major reasons is that data centers now consume 3% of the electricity in the United States as per an estimate by the Environmental Protection Agency (EPA). Companies' endeavour is to explore and harness renewable sources of the nature for cooling their data centers. Tang *et al.* proposed a heat recirculation model and developed an algorithm called MPIT-TA to reduce cooling cost by minimizing the peak inlet temperature [19]. To reduce the on-chip temperature, Chaparro *et al.* investigated the correlation between processor utilization and temperature increases in processors [1]. Chaudhry *et al.* presented an array of thermal-ware scheduling policies tailored for green data centers, where heat modeling, thermal-aware monitoring and profiling are integrated [2]. Kaushik and Bhandarkar developed GreenHDFS, where a Hadoop cluster is divided into Hot and Cold zones [9]. A hot zone is a collection of nodes currently executing jobs, whereas a cold zone has nodes sitting idle in low-power mode. A zone's temperature is determined by its power consumption and the data is placed onto a suitable temperature zone by using data classification policies [9]. Our study is different from the aforementioned research in essence that we focus on the thermal behaviors of both processors and disks in the realm of cluster computing.

2.2 Job Schedulers

A large number of novel schedulers were proposed to optimize the energy and thermal efficiency of single machines [4][11]. Job schedulers play a vital role in improving energy efficiency and performance of large-scale clusters [3][17][16]. Very recently, an increasing number of novel schedulers have been proposed to Hadoop clusters housed in data centers. For example, Pastorelli *et al.* designed a job scheduler called HFSP for Hadoop clusters. HFSP implements a size-based preemptive scheduling principle, which aims at reducing overall system response times and guaranteeing fairness by building job size information using virtual time and aging as metrics [15]. Zaharia *et al.* proposed a mechanism, *Delay Scheduling*, that achieves nearly 100% locality while slightly relaxing fairness. The *delay-scheduling* idea was integrated into Hadoop's fair scheduler to improve the response time of small jobs by a factor of five (5) and to double throughput in I/O-heavy workload [22]. Radheshyam *et al.* proposed a scheduling algorithm that involves task selection and task assignment to choose the best task suitable for a particular data node. Their algorithm applies heuristic and machine learning solutions to balance resources on clusters to reduce overall runtime of submitted jobs [14]. Zaharia *et al.* also developed a scheduler, LATE, which uses estimated finish times to speculatively execute those tasks that degrade the response time [23]. LATE outperforms Hadoop's default speculative execution algorithm in real workloads on Amazon's Elastic Compute Cloud. Mashayekhy *et al.* proposed two energy-efficient MapReduce scheduling algorithms (i.e., *EMRSA*) assigning map and reduce tasks to appropriate nodes in order to minimize energy consumed in Hadoop clusters [12].

Unlike the above performance-oriented schedulers, we propose a thermal-aware scheduler that dispatches Hadoop jobs based on their profiled thermal characteristics. While most of the existing Hadoop schedulers are improving overall cluster performance, our *tDispatch* is conducive of optimizing thermal

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.

efficiency and reducing cooling cost of Hadoop clusters in data centers.

3 Thermal Profiling

To understand and develop a relationship among thermal impacts of different types of jobs on a Hadoop cluster, we study the thermal behavior of master and data nodes on a homogeneous Hadoop cluster, where all the nodes share an identical configuration (1). After summarizing the experiment setups (see Section 3.1), we discuss the thermal behaviors of various MapReduce benchmarks (see Section 3.2).

3.1 Experimental Testbed

Hadoop Server

In this group of experiments, we observe the thermal behaviors of a homogeneous cluster of 15 nodes (labelled as *backend-1-0 to backend-1-16*) placed on one rack in our HPC room (*backend-1-4* and *backend-1-5* were not available for this experiment). The room temperature is set to 20°C Table 1 describes the configuration of these nodes.

Hardware	
Computer	SuperMicro Model 825-7
CPU	Intel (R) Xeon (R) X5650@2.67GHz (cores: 24)
Memory	24 GB DDR3 SDRAM at 1066 MHz
Network controller	GigaBit Ethernet and 20 Gb InfiniBand
Disks	500GB
Software	
Operating System	Cent OS 7.2 64-bit
Hadoop	Release 2.7.3
Linux Kernel	3.10.0

TABLE 1: Hardware and software configurations of the testbed

MapReduce is a framework for processing parallelizable problems across a multitude of computing nodes, collectively referred to as a cluster (if all nodes are on the same local network and use similar hardware) or a grid (if the nodes are shared across geographically and administratively distributed systems, and use more heterogenous hardware) [20]. Furthermore, many MapReduce applications show significant variation in resource (CPU/disks/DRAM) consumption during their running time. MapReduce applications range from CPU-intensive to I/O-intensive, and the workload of each subtask is either CPU-bound, I/O-bound, or network-bound. To investigate the variation in their behavior, we run a mix of I/O-intensive and CPU-intensive MapReduce jobs on the homogeneous Hadoop cluster. An I/O-intensive application spends a significant portion of the time processing I/O operations rather than computing tasks. A CPU-intensive application, in contrast, infrequently generates I/O requests while spending a vast majority of time doing computations [18]. We choose a few micro benchmarks from a popular MapReduce benchmark suite called *HiBench* [8].

We choose four popular Hadoop benchmarks, namely, WordCount, PageRank, Sort, and TeraSort and we vary the input data sizes (tiny, small, large, huge, and gigantic) for each of these four benchmarks as available with *HiBench* benchmark suite [8]. To measure processor temperatures, we use a Linux utility program called *lm-sensors*; we apply another utility called *hddtemp* to monitor internal hard drive temperatures; we make use of the *iostat* tool to measure CPU and disk utilization. CPU utilization represents the computing load across



Fig. 1: Cluster Setup in High Performance Computing Lab at Auburn University (Only two racks shown)

HiBench Input Size	Megabytes (Mb)
Tiny	3200
Small	320000000
Large	3200000000
Huge	32000000000
Gigantic	320000000000
Big Data	6000000000-1600000000000

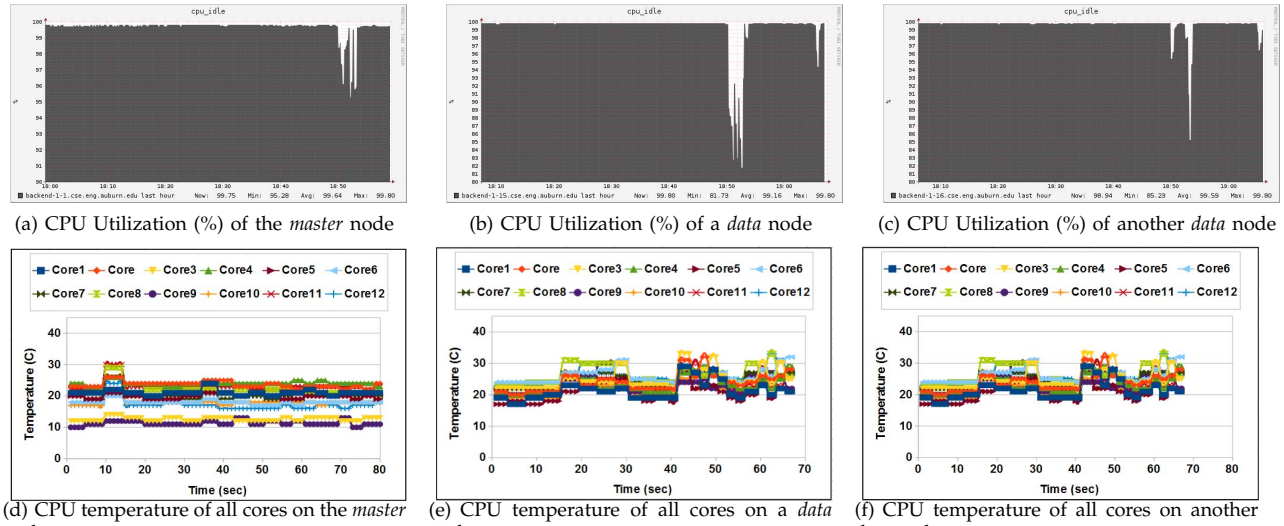
all the four cores, whereas disk utilization represents I/O load imposed by tested Hadoop applications. Further, we use *Ganglia*, a monitoring system for high-performance computing systems like clusters. We use the elapsed wall clock time as execution time. We emphasize here that our work is comprised of measuring actual execution time, CPU/disk temperatures, and utilization of the aforementioned MapReduce applications without relying on any simulation data.

The following factors inspire us to investigate thermal behaviors of homogeneous clusters without addressing the air re-circulation and node placement issues.

- In this study, we focus on homogeneous clusters, because nodes in a brand new cluster are typically homogeneous in nature. It is worth mentioning that our thermal profiling research can be extended to heterogeneous clusters.
- For the purpose of measurements, the evaluated nodes are placed in a traditional rack. The node placement in a rack, of course, can affect air re-circulation in a data center; nevertheless, air re-circulation is not investigated in our study. Our thermal profiling results can be incorporated into a data center's thermal model, where air re-circulation is taken into account.

In the following sections, we will analyze the thermal behavior of our cluster running Hadoop with the aid of graphs obtained from *Ganglia*.

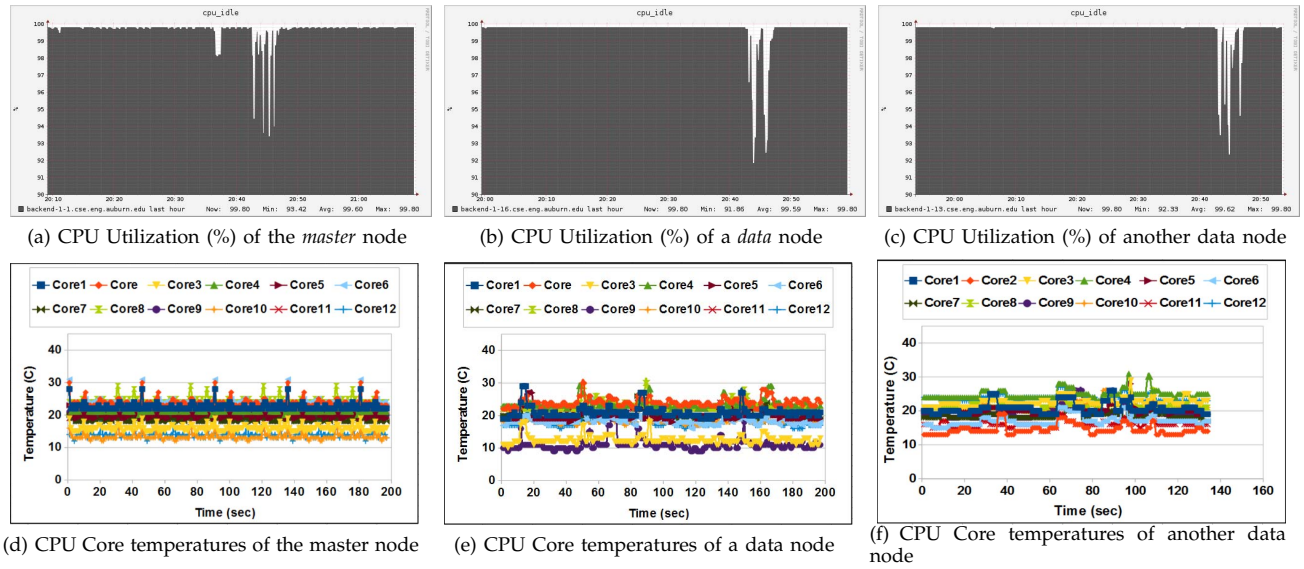
Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.



(a) CPU Utilization (%) of the *master* node (b) CPU Utilization (%) of a *data* node (c) CPU Utilization (%) of another *data* node

(d) CPU temperature of all cores on the *master* node (e) CPU temperature of all cores on a *data* node (f) CPU temperature of all cores on another *data* node

Fig. 2: The figures above demonstrate the general behavior of the WordCount benchmark.



(a) CPU Utilization (%) of the *master* node (b) CPU Utilization (%) of a *data* node (c) CPU Utilization (%) of another data node

(d) CPU Core temperatures of the master node (e) CPU Core temperatures of a data node (f) CPU Core temperatures of another data node

Fig. 3: Average CPU utilization and temperatures of all cores of the master and two data nodes on the Hadoop cluster running TeraSort benchmark.

3.2 Thermal Behaviors of the Benchmarks

In this section, we study the thermal behaviors of four MapReduce benchmarks applications, namely, WordCount, Sort, PageRank, and TeraSort.

3.2.1 WordCount

The tested WordCount is a MapReduce application running on a Hadoop cluster, where text files are loaded and the frequency of each word in the files is counted. The input to Wordcount is text file(s), and output is also a text file containing words followed by number of times they occur. By the nature of it, WordCount is a CPU-bound application as it spends most of its time in computing.

The purpose of using WordCount is to study the thermal properties of the processors of the Hadoop cluster's nodes by keeping CPU extremely busy. We make WordCount process a huge-sized data (i.e., 32G) available through *HiBench* in order to make the experiment run for a long duration of time, which helps us to better understand of the cluster's behavior.

Fig. 2(a) shows the CPU utilization of all the cores residing in the master node. We observe that CPU utilization of the master node change insignificantly; the disk temperature stays almost as a constant (not presented here). This thermal trend is expected, because the master node is simply responsible for dispatching MapReduce jobs to the data nodes without dealing with any CPU-bound or I/O-bound tasks. For example, Fig. 2(d) confirms that the CPU temperature of the master node is extremely low.

We measure and collect thermal trends of the data node(s) that ran the Wordcount. Due to space limitation and the similar

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.

trends of all the data nodes, we only present the thermal results of the two nodes in Figs. 2(e) and 2(f), each of which depicts the temperatures of the twelve cores as functions of time. Figs. 2(e) and 2(f) show that the temperatures of all the cores increase gradually. Not surprisingly, the data nodes' CPU temperatures are much higher than those of the master node (see also Fig. 2(d)), because most of the work is allocated to the data nodes. Moreover, we noticed in our results that there is not much deviation in CPU core temperatures inside the same chip. The average CPU temperature of all the multiple cores in one data node (see Fig. 2(e)) is 28.5°C. The CPU temperatures start diminishing towards the end of the experiments on all the data nodes.

We observed that compared to the processor temperatures, disk temperatures are insensitive to the WordCount workload. More specifically, the disk temperatures rise very slowly as the experiment proceeded. It takes around 15-20 minutes for the disks to heat up by 3°C on the slave nodes. Since the disks are not as thermally sensitive to workload as processors, no cooling down phase of the disks is observed immediately at the end of the experiments. For the same reason, we do not present the disk temperature here and thus, focusing only on CPU temperatures.

Figs. 2(a), 2(b), and 2(c) show the CPU utilization patterns on master and data nodes of our cluster running WordCount. We observe that for a slight increase in CPU utilization on slave nodes, the CPU temperature shoots up significantly (see Figs. 2(e) and 2(f)).

3.2.2 TeraSort and Sort

We make use of TeraSort - a MapReduce benchmark - to measure the execution time spent in sorting terabytes of data on a Hadoop cluster. The performance of TeraSort depends on the amount of available resources on the cluster. Unlike WordCount, TeraSort spends a significant amount of time performing input-output operations; we classify TeraSort as an I/O-intensive application.

We make TeraSort process a large-sized data available with HiBench benchmark suite (i.e., 32G). Unsurprisingly, we observe similar trends in CPU temperature among all the data nodes and; therefore, we select the data node which was deployed to run TeraSort and the results are shown in this subsection. We have represented CPU utilization and temperature trends here. The input data of TeraSort are partitioned equally among the available cores; therefore, all the cores on each node exhibit identical heat patterns. We observe interesting heat patterns during the execution of TeraSort. We present the CPU utilization patterns for the master node (see fig. 4(a)) and two representative data nodes (see fig. 4(b) and fig. 4(b)). We observe that the CPU utilization is significantly low for disk-intensive jobs. Fig. 3(d) shows the temperature of the twelve cores and on the master node. Because the master node allocates a majority of the work to the data nodes, the CPU utilization remains at a low level on the master. As such, the corresponding temperatures are also remain low. As the data nodes starts processing the job assigned to them by the master node, the CPU temperatures on the slave nodes start rising due to the extensive disk usage. The CPU temperature rises occasionally but remains steady for most of the time (see Figs. 3(e) and 3(f)).

Similar trends for *Sort* are observed but in the interest of space, the figures are not represented here. The CPU utilization as well as the CPU temperatures stay very minimal with running huge-sized Sort. We run ten iterations of Sort to better understand it's thermal behavior.

3.2.3 PageRank

Now we test PageRank, which is a open source implementation of page-rank algorithm and is used in web search engines that calculate the ranks of web pages according to the numbers of references links [8]. This is helpful in finding the most relevant information for the searched keyword(s). Page ranking in Hadoop involves parsing, calculating, and ordering.

PageRank spends considerable amount of time on iterating small jobs and most of these jobs are CPU bound, with some memory and disk I/O consumption. The goal of this group of experiments is to study PageRank's behaviors. Compared to its counterparts (i.e., WordCount, Sort, and TeraSort), PageRank has the longest execution time given the same size of input data. Thus, it is an ideal candidate to examine the thermal behaviors of the cluster in a long interval. We set the PageRank's input data size to 32GB (which is categorized as *huge* as per *HiBench*'s default configuration) and run it ten time to understand the behavior clearly. Similar to the previous experiments, we only present the results of the three data nodes; disk utilization trends are omitted in this subsection.

Figs. 4(a), 4(b), and 4(c) show the CPU utilization of the master and two data nodes. Fig. 4(d) shows the temperatures of all 12 cores residing on the master node. The CPU thermal trends are expected, because the role of the master node is to dispatch all the tasks to the data nodes.

Figs. 4(e) and 4(f) demonstrate the behavior of PageRank on the data nodes, where we observe an interesting pattern of the CPU temperatures. The CPU temperatures start rising as the experiment progresses and remain considerable high throughout the experiment. The maximum CPU core temperature on one of the data nodes is 39°C (see Fig. 4(e)) while the maximum disk temperature on another data node is 38°C (see Fig. 4(f)).

4 A Thermal-aware Job Scheduler

Now we propose *tDispatch* - a thermal-aware job scheduler aiming to reduce the CPU and disk temperatures of Hadoop clusters.

During the process of designing *tDispatch*, we answer the following two questions:

- How should we classify Hadoop jobs into CPU-intensive and I/O-intensive categories? (see Section 4.1)
- How do we make use of these two job categories to improve thermal management of Hadoop clusters? (see Section 4.2)

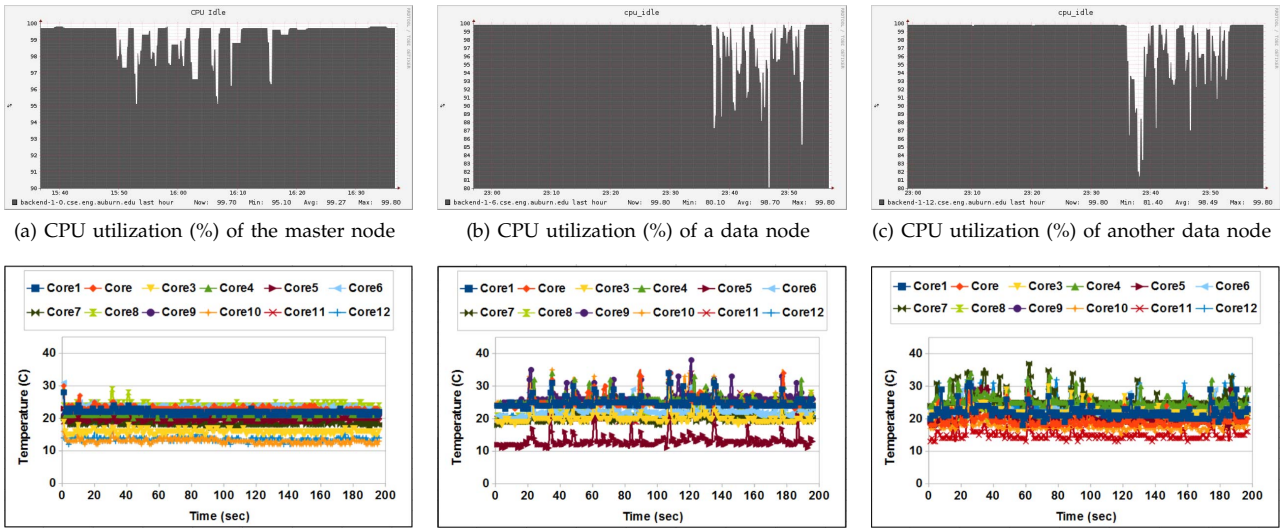
4.1 Job Profiling

We run and profile WordCount and TeraSort on a Hadoop cluster. We only run one job at a time and collect the CPU and disk temperatures of the cluster nodes.

Using the results of our prior experiments, we identify the access patterns of MapReduce applications by taking resource utilization of subtasks into account. Since each subtask issues either computation or I/O requests, we can categorize these benchmarks in one of the two categories: CPU-Intensive or I/O-intensive.

We classify Hadoop jobs into the CPU-intensive and I/O-intensive ones, because CPU and disk temperatures are highly sensitive to CPU and disk utilization, respectively. In what

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.



(a) CPU utilization (%) of the master node (b) CPU utilization (%) of a data node (c) CPU utilization (%) of another data node

(d) CPU temperature of the master node (e) CPU temperature of a data node (f) CPU temperature of another data node

Fig. 4: Average CPU utilization and temperatures of all cores of the master and two data nodes on the Hadoop cluster running *PageRank*

follows, we show how to rely on profiling information to determine if a job is a CPU-intensive or I/O-intensive one. The *tDispatch* makes use of this information to dispatch jobs in a thermal-friendly way. We also show that CPU-intensive jobs have significant thermal impacts on multicore processors, whereas I/O-intensive jobs noticeably increase disk temperatures of Hadoop clusters.

4.2 Job Scheduler Design

In this part of the study, we design a thermal-aware job scheduling strategy aimed to keep CPU and disk temperatures of cluster nodes as low as possible. We design *tDispatch* - a job scheduler tailored for big-data applications running on large-scale Hadoop clusters deployed in data centers. Hadoop's fair scheduling policy assigns a fair share of CPU resources to all running jobs with minimum starvation. *tDispatch* addresses the downside of the existing job schedulers that overlook thermal efficiency, the improvement of which has become a critical concern for data centers.

4.2.1 Two Modules

Our job scheduling mechanism consists of two key components, namely, *tQueue* and *tDispatch*. The *tQueue* module is responsible for maintaining two dedicated queues - one for CPU-intensive jobs (i.e., Q^c) and one for I/O-intensive jobs (i.e., Q^d). The master node receives incoming jobs through clients. The received jobs are appended to one of the two queues based on their types. If a job is CPU-intensive, it is queued up into Q^c; otherwise, the I/O-intensive job is queued up in Q^d.

Algorithm 1 presents the pseudocode of *tQueue*. It is noteworthy that *tQueue* has high extensibility, because *tQueue* can be readily extended by managing additional queues for memory-intensive jobs or jobs with other types of characteristics.

We design *tDispatch* (see Algorithm 2) to allocate resources of a Hadoop cluster to one of the jobs in queues Q^c and Q^d, which are managed by *tQueue*. After a running job finishes its execution, *tDispatch* thermal-efficiently picks another candidate job from Q^c and Q^d to run on the cluster. In our design, the interface between *tQueue* and *tDispatch* are Q^c and Q^d.

Algorithm 1: tQueue. **Input:** Job J_i; **Output:** Q^c, Q^d.

```
if J_i.type = I/O − intensive then
    Q^d.enqueue(J_i);
else
    Q^c.enqueue(J_i); /*job j is CPU-intensive*/
end if
```

Algorithm 2: tDispatch. **Input:** Q^c, Q^d.

```
Initialize W_c ← 0, W_d ← 0
while a job finishes its execution do
    if i.t_c < R_c then
        if i.t_d < R_d then
            dispatch a job from Q^c and Q^d using FIFO;
        else if i.t_d > R_d and W_d < S then
            dispatch a job from Q^c to node i;
            W_c ← 0;
            W_d ← W_d + 1;
        else
            dispatch a job from Q^d to node i;
            W_d ← 0
        end if
    else if i.t_c > R_c then
        if W_c < S then
            dispatch a job from Q^d;
            W_d ← 0;
            W_c ← W_c + 1;
        else
            dispatch a job from Q^c;
            W_c ← 0;
        end if
    else
        dispatch a job from Q^c and Q^d using FIFO;
    end if
end while
```


Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.

4.2.2 Fairness and Starvation Avoidance

One novelty of our design is that *tDispatch* cools down CPU and disk temperatures by dispatching two types of jobs - CPU-intensive and I/O-intensive ones. To avoid starvation and guarantee fairness between these two types of jobs, we ensure that the number of consecutive runs of the same type of jobs is always below an upperbound (see S in Lines 6 and 15 of Algorithm 2). In doing so, no job will be starved in the queues. This fairness-maintaining idea is implemented by introducing two counters W_c and W_d (see Line 1 of Algorithm 2), which keep track of the number of times one type of jobs' execution has been postponed. For example, if the scheduler repeatedly dispatches CPU-intensive jobs from queue Q_c to reduce disk temperatures, then counter W_d is increased by 1 after each CPU-intensive job is dispatched (see Lines 6-9 of Algorithm 2). When counter W_d reaches the upperbound S, an I/O-intensive job must be dispatched regardless of disk temperatures (see Lines 10-13 of Algorithm 2). In this case, counter W_d is reset to 0 after the I/O-intensive job starts its execution, thereby offering additional opportunities to continue cooling down disks.

4.2.3 Thermal Awareness

During the process of choosing the best job candidate from the queue Q_c and Q_d, algorithm *tDispatch* checks if the average CPU and disk temperatures of the Hadoop cluster are well below redline temperatures, which are recommended by manufacturers to safely operate devices. Here we denote R_c and R_d as the threshold temperatures of CPUs and disks. When the cluster's average CPU temperature exceeds threshold R_c, *tDispatch* starts running I/O-intensive jobs rather than CPU-intensive ones to lower CPU temperatures.

tDispatch makes an effort to keep CPU and disk temperatures below thresholds R_c and R_d. *tDispatch* achieves this goal by taking advantage of the fact that a CPU-intensive job tend to further raise the CPU temperatures whereas an I/O-intensive job inevitably augment the disk temperature. Therefore, when CPU temperature is nearly equal to temperature threshold R_c, *tDispatch* postpones the execution of any CPU-intensive job by dispatching I/O-intensive jobs in queue Q_d. Consequently, *tDispatch* keeps the overall CPU temperature low (see Lines 14-20 of Algorithm 2).

Similarly, *tDispatch* attempts to reduce overall disk temperatures (see Lines 6-9 of Algorithm 2). Specifically, if the cluster's average disk temperature goes beyond threshold R_d, then CPU-intensive jobs rather than I/O-intensive ones are dispatched to the cluster.

4.2.4 Two Extreme Scenarios

If both average CPU and disk temperatures are below threshold temperatures R_c and R_d, then our *tDispatch* scheduler follows the *FIFO* policy to pick a job with the earliest arrival time from queues Q_c and Q_d, and dispatches the chosen job to the Hadoop cluster for execution (see Lines 3-5 of Algorithm 2).

Under heavy workload conditions, both CPU and disk temperatures are likely to exceed threshold R_c and R_d. In this extreme case, *tDispatch* applies *FIFO* to schedule jobs in queues Q_c and Q_d (see Line 24 of Algorithm 2).

4.3 Extensibility

Since our cluster is homogeneous in nature, the CPU and disk threshold temperatures (i.e., R_c and R_d) are identical across all the nodes. Nevertheless, our algorithm can be extended to handle the heterogeneous case where multiple nodes have various CPU and disk redline temperatures. When it comes to heterogeneous clusters, we replace threshold temperatures R_c and R_d with two threshold vectors $\hat{R_c}$ and $\hat{R_d}$, where different nodes have distinct threshold temperatures maintained in the two vectors.

In the aforementioned two extreme scenarios (i.e., very light or heavy load, see Section 4.2.4), *tDispatch* applies the *FIFO* policy. Our algorithm *tDispatch* is extensible in a way that *FIFO* can be readily substituted by any other policy that optimizes overall system performance. For example, if we adopt *tDispatch* to deal with real-time jobs, a good scheduling policy to replace *FIFO* is the earliest deadline first or *EDF*.

5 Experimental Results

In this section, we evaluate the thermal efficiency and performance of our proposed thermal-aware job scheduling strategy (i.e., *tDispatch*). The goal of our experiments is to demonstrate that the tDispatch scheduler is capable of improving thermal efficiency of Hadoop clusters by the virtue of its thermal awareness.

5.1 CPU-intensive and I/O-intensive Jobs

We are focusing on two popular types of Hadoop jobs, namely, CPU-intensive and I/O-intensive jobs. The tested Hadoop jobs were implemented using the MapReduce programming model, where each MapReduce application comprises of a large number of mapper and reducer tasks running on clusters. These mappers and reducers may be either CPU-bound, I/O-bound, or network bound. CPU-intensive jobs tend to increase processor temperatures, whereas I/O-intensive jobs noticeably affect disk temperatures. We choose WordCount and PageRank as the CPU-intensive jobs; we also run TeraSort and Sort as representatives of I/O-intensive jobs.

For fair comparison, we set the size of data for each job, regardless of its type, to be around HiBench-large. The duration of these jobs is short and this short execution time resemble real-world cloud computing environments where a time-sharing policy is enforced.

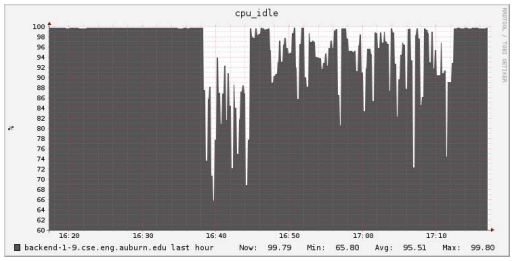
With our scheduler in place, system administrators can easily maintain the overall temperature of a Hadoop cluster below a specified redline temperature without lowering the supply temperature of a cooling system's. Hence, our thermal-aware scheduler offers an energy-saving capability through improving the cooling system's energy efficiency.

5.2 Baseline Schedulers

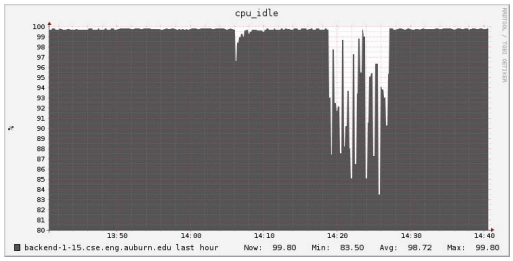
To demonstrate the thermal awareness of tDispatch, we compare our tDispatch with two baseline schedulers. The first one called *IOF* (i.e., I/O-intensive job first) gives high priority to I/O-intensive jobs; the second scheduler named *CPUF* (i.e., CPU-intensive job first) allows CPU-intensive jobs to be executed prior to any I/O-intensive ones. These two schedulers favor different types of Hadoop jobs submitted in a queue of the evaluated Hadoop cluster. We compare the thermal effectiveness of our tDispatch with those of IOF and CPUF.

In addition to evaluating the thermal behavior of tDispatch, we investigate whether favoring CPU-intensive jobs is more thermal friendly than favoring I/O-intensive jobs on Hadoop clusters or vice versa. In one of our future studies, we will compare tDispatch other candidate schedulers including fair scheduler and FIFO other than IOF and CPUF.

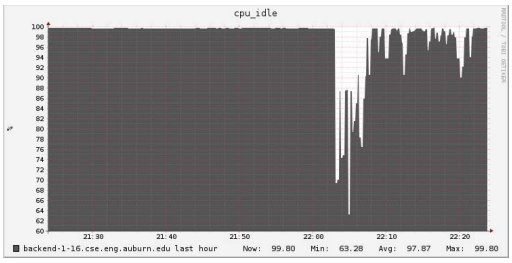
Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.



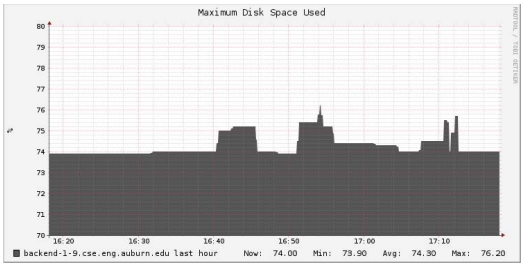
(a) CPU utilization (%) of a data node with CPUF



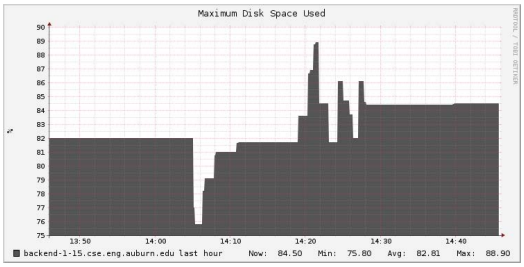
(b) CPU utilization (%) of a data node with IOF



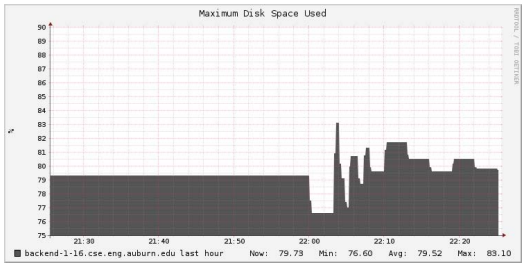
(c) CPU utilization (%) of a data node with *tDispatch*



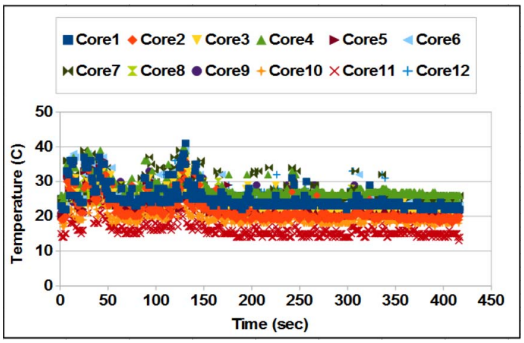
(d) Disk utilization (%) of a data node with CPUF



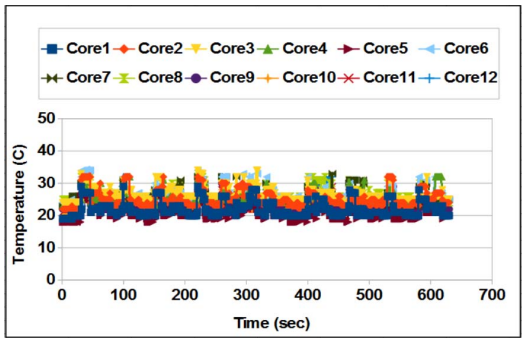
(e) Disk utilization (%) of a data node with IOF



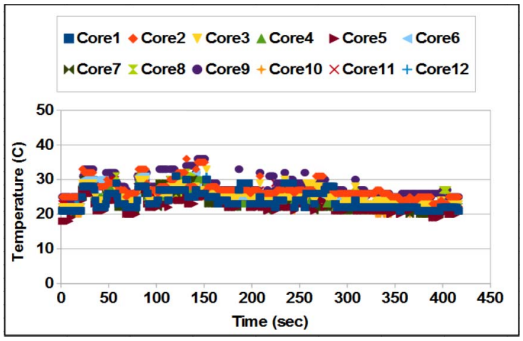
(f) Disk utilization (%) of a data node with *tDispatch*



(g) CPU core temperature of a data node with CPUF



(h) CPU core temperatures of a data node with IOF



(i) CPU core temperatures of a data node with *tDispatch*

Fig. 5: Average CPU and disk temperatures of master node on a Hadoop cluster running Terasort, WordCount, PageRank, and Sort with different schedulers: CPUF, IOF, and *tDispatch*.

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.

5.3 Experiment Methodology

In each test, we submit a total of 20 *HiBench-large* sized CPU- and I/O-intensive jobs to our Hadoop cluster, where a dispatch queue (a.k.a., batch queue) maintains all the submitted jobs. In our experiments, two CPU-intensive jobs are *WordCount* and *PageRank*; two I/O-intensive jobs including *Terasort* and *Sort*. After all the jobs are submitted to the batch queue, our scheduler picks an appropriate job to be launched on the Hadoop cluster. When the running job completes its execution, the scheduler choose the next most appropriate job to be running on the cluster.

We keep track of the CPU and disk temperatures of the master and data nodes in the Hadoop cluster. The temperature measurements allow us to quantitatively assess the thermal-efficiency improvement offered by *tDispatch* from the perspective of both CPU and disk resources.

In addition to collecting CPU and disk temperatures, we monitor CPU and disk utilization during the course of running the submitted jobs. In the interest of space, we did not present disk temperatures since disks take a considerable amount of time to heat up and thus do not show interesting trends. We intend to show that tDispatch is able to reduce system temperatures without making any adverse impact on the performance of Hadoop clusters.

5.4 Results and Discussions

5.4.1 Master Node

In the first group of experiments, we measure the CPU and disk temperatures of the master node governed by the three schedulers. We observe that the schedulers have little impact on the thermal efficiency of the master node. Comparing Fig. 5 with Figs. 2(a) and 3(a) found in Section 3.2, we conclude that both Hadoop jobs and the scheduler make no noticeable thermal impact on the master node. Because the master node is not a thermal-efficiency bottleneck, the results suggest that we should pay attention to improve thermal efficiency of data nodes rather than a master node.

5.4.2 Data Nodes

In this group of experiments, we compare the thermal impacts of the three tested schedulers on the Hadoop cluster from the perspective of data nodes. All the data nodes share a similar thermal-efficiency trend and; therefore, we pick one data node as a representative to demonstrate the thermal behaviors of the data nodes.

Figs. 5(a), 5(b), and 5(c) show the CPU and disk utilization of the nodes when the jobs are running with these three scheduling policies. We observe that the average CPU utilization for the representative nodes with both CPUF and IOF remain considerably high during the execution of the jobs; tDispatch manages to reduce the overall average of CPU utilization (see 5(c)). CPU utilization is in direct correlation with the power consumption; tDispatch is conducive of saving energy. We observe that the disk utilization is considerably reduced with *tDispatch* (Fig:5(f)) as opposed to CPUF and IOF (figs. 5(d) and 5(e)).

Figs. 5(g), 5(h), and 5(i) keep track of the temperatures of all twelve cores in the data node for our three chosen schedulers. We measure the average CPU temperature of each node during the course of the experiments. Instead of calculating the average temperatures across all the data nodes, we focus on the CPU utilization and the peak temperature reached since the three experiments last for a different duration in time. Instead of using average disk utilization, we observe the rise in disk utilization for all the three cases.

Our *tDispatch* is able to bring down the peak temperature of these nodes. The disk temperatures in all the three scheduler cases remain flat mainly because of the relatively low I/O load and short observation time period. Nevertheless, *tDispatch* noticeably reduces the average disk temperatures of CPUF and IOF approximately by 1°C (figure not presented here).

5.5 Reduction in Peak Temperature and Energy-efficiency

When it comes to CPU thermal efficiency, *tDispatch* is superior to CPUF and IOF as it can prevent hot spots by demonstrating that maximum difference in temperature that any core had from the average temp at each iteration is lower and optimizing the peak temperature to reduce the overall power consumption. For example, the peak CPU temperature of the representative node is governed by tDispatch is 36.2°C (see fig. 5(g)) , whereas the average CPU temperatures of the same node through CPUF and IOF are 41°C and 34.7°C, respectively.

To estimate the energy efficiency of our scheduler, we used network power distribution unit (PDU) installed in our HPC room to collect the energy consumption by our nodes. An HPC application's power and energy consumption can vary during the lifetime of the application due to the nature of application itself. We use the mean power to compute energy consumption for four different jobs with each of our schedulers. We ran three set of experiments here by varying the input data size for our four chosen applications; the experiments finished at almost the same time and this helps us with a fair comparison among all three schedulers in each size category.

We notice that the energy consumption of very small jobs (labeled as *tiny* in HiBench) is relatively high when compared with other two job sizes. Out of the three cases, the energy consumption is minimal with small size input for the four applications. Regardless of the job size, tDispatch is able to mitigate the energy consumption for these jobs. From the energy consumption, we can derive the cooling cost of the data centers running these jobs.

6 Future Research Directions

As a future research direction, we plan to extend *tDispatch* to meet the needs of time-sharing Hadoop platforms facilitating cloud services. We will develop a thermal model to predict outlet temperatures of data nodes of Hadoop clusters. We intend to seamlessly integrate the thermal model with the proposed *tDispatch* scheduler, thereby reducing the cooling cost of data centers.

A second intriguing research direction is to consider other job categories other than the CPU-intensive and I/O-intensive ones. For example, we will design a thermal-aware scheduler tailored for both batch jobs and interactive jobs. Combining the batch and interactive job categories with the CPU-intensive and I/O-intensive camps, we form four job classes - (1) CPU-intensive batch, (2) CPU-intensive interactive, (3) I/O-intensive batch, and (4) I/O-intensive interactive jobs. We plan to look into the behaviors of other popular benchmark categories also included in HiBench, namely *Web Search, Machine Learning, and HDFS Benchmark*.

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.

7 Conclusions

Recognizing that improving thermal efficiency of clusters helps in reducing the operational cost of data centers, we developed a thermal-aware job scheduling strategy called *tDispatch* for MapReduce applications running on Hadoop clusters. To facilitate the development of *tDispatch*, we conducted a profiling study of CPU-intensive and I/O-intensive jobs from the perspective of thermal efficiency. We investigated the thermal behaviors of CPU-intensive and I/O-intensive applications running on Hadoop clusters by stress testing data nodes through extensive experiments. Our profiling results show that CPU-intensive and I/O-intensive jobs impose distinctive thermal and performance impacts on multicore processors and hard drives of Hadoop clusters.

After we offered detailed analysis on the thermal profiles of the Hadoop applications, we elaborated the design of the *tDispatch* scheduler, which judiciously dispatches CPU-intensive and I/O-intensive jobs to Hadoop clusters in a way to keep CPU and disk temperatures low.

Our scheduler consists of two key components - *tQueue* and *tDispatch*. The *tQueue* module is responsible for maintaining a CPU-intensive-job and I/O-intensive-job queues. We implemented *tDispatch* to allocate resources of a Hadoop cluster to one of the jobs in these two queues. After a running job finishes its execution, *tDispatch* thermal-efficiently picks another candidate job from the job queues to run on the cluster. Our scheduler alternatively chooses CPU-intensive and I/O-intensive jobs from the two queues to run on a Hadoop cluster, offering ample opportunities for multicore processors and disks to cool down.

We conducted an empirical study to compare the thermal performance of *tDispatch* with two scheduler counterparts, namely, CPUF and IOF. Specifically, we measured the CPU and disk temperatures of the master node and data nodes governed by the three schedulers. Our findings suggest that the schedulers have little impact on the thermal efficiency of the master node. More importantly, our experimental results show that *tDispatch* is superior to CPUF and IOF in terms of improving thermal efficiency of data nodes.

Acknowledgment

This research was supported by the U.S. National Science Foundation under Grants CCF-0845257 (CAREER), CNS-0917137 (CSR), CNS-0757778 (CSR), CCF-0742187 (CPA), CNS-0831502 (CyberTrust), CNS-0855251 (CRI), OCI-0753305 (CI-TEAM), DUE-0837341 (CCLI), and DUE-0830831 (SFS).

References

[1] P. Chaparro, G. Magklis, J. Gonzalez, and A. Gonzalez, "Distributing the frontend for temperature reduction," in *High-Performance Compthermalmodeluter Architecture, 2005. HPCA-11. 11th International Symposium on*, Feb 2005, pp. 61–70.

[2] M. T. Chaudhry, T. C. Ling, A. Manzoor, S. A. Hussain, and J. Kim, "Thermal-aware scheduling in green data centers," *ACM Computing Surveys (CSUR)*, vol. 47, no. 3, p. 39, 2015.

[3] ——, "Thermal-aware scheduling in green data centers," *ACM Comput. Surv.*, vol. 47, no. 3, pp. 39:1–39:48, Feb. 2015. [Online]. Available: http://doi.acm.org/10.1145/2678278

[4] A. Che, K. Lv, E. Levner, and V. Kats, "Energy consumption minimization for single machine scheduling with bounded maximum tardiness," in *Networking, Sensing and Control (ICNSC), 2015 IEEE 12th International Conference on*. IEEE, 2015, pp. 146–150.

[5] J. Donald and M. Martonosi, "Techniques for multicore thermal management: Classification and new exploration," in *Computer Architecture, 2006. ISCA '06. 33rd International Symposium on*, 2006, pp. 78–88.

[6] K. Fowler, "Hadoop forensics, tackling the elephant in the room," in *Security Education Conference*, 2012.

[7] O. V. Geet, "Trends in data center design - ashrae leads the way to large energy savings," in *T*, 2014.

[8] S. Huang, J. Huang, J. Dai, T. Xie, and B. Huang, "The hibench benchmark suite: Characterization of the mapreduce-based data analysis," in *Data Engineering Workshops (ICDEW), 2010 IEEE 26th International Conference on*. IEEE, 2010, pp. 41–51.

[9] R. T. Kaushik and M. Bhandarkar, "Greenhdfs: towards an energy-conserving, storage-efficient, hybrid hadoop compute cluster," in *Proceedings of the USENIX Annual Technical Conference*, 2010, p. 109.

[10] F. Kong and X. Liu, "A survey on green-energy-aware power management for datacenters," *ACM Computing Surveys (CSUR)*, vol. 47, no. 2, p. 30, 2015.

[11] J. Li, M. Qiu, J.-W. Niu, L. T. Yang, Y. Zhu, and Z. Ming, "Thermal-aware task scheduling in 3d chip multiprocessor with real-time constrained workloads," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 12, no. 2, p. 24, 2013.

[12] L. Mashayekhy, M. M. Nejad, D. Grosu, D. Lu, and W. Shi, "Energy-aware scheduling of mapreduce jobs," in *Proceedings of the 2014 IEEE International Congress on Big Data*, ser. BIGDATACONGRESS '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 32–39. [Online]. Available: http://dx.doi.org/10.1109/BigData.Congress.2014.15

[13] J. Moore, R. Sharma, R. Shih, J. Chase, R. Patel, P. Ranganathan, and H. P. Labs, "Going beyond cpus: The potential of temperature-aware solutions for the data center," in *In Proceedings of the Workshop of Temperature-Aware Computer Systems (TACS-1) held at ISCA*, 2004.

[14] R. Nanduri, N. Maheshwari, A. Reddyraja, and V. Varma, "Job aware scheduling algorithm for mapreduce framework," in *Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on*, Nov 2011, pp. 724–729.

[15] M. Pastorelli, A. Barbuzzi, D. Carra, M. Dell'Amico, and P. Michiardi, "Hfsp: Size-based scheduling for hadoop," in *Big Data, 2013 IEEE International Conference on*, Oct 2013, pp. 51–59.

[16] X. Qin, H. Jiang, A. Manzanares, X. Ruan, and S. Yin, "Dynamic load balancing for i/o-intensive applications on clusters," *ACM Transactions on Storage (TOS)*, vol. 5, no. 3, p. 9, 2009.

[17] ——, "Communication-aware load balancing for parallel applications on clusters," *Computers, IEEE Transactions on*, vol. 59, no. 1, pp. 42–52, 2010.

[18] A. Silberschatz, *Operating System Concepts, 9th Edition*. John Wiley Sons, 2012.

[19] Q. Tang, S. Gupta, and G. Varsamopoulos, "Energy-efficient thermal-aware task scheduling for homogeneous high-performance computing data centers: A cyber-physical approach," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 19, no. 11, pp. 1458–1472, Nov 2008.

[20] Wikipedia. (2016) Mapreduce. [Online]. Available: https://en.wikipedia.org/wiki/MapReduce

[21] H. Xu, C. Feng, and B. Li, "Temperature aware workload managementin geo-distributed data centers," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 26, no. 6, pp. 1743–1753, 2015.

[22] M. Zaharia, D. Borthakur, J. Sen Sarma, K. Elmeleegy, S. Shenker, and I. Stoica, "Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling," in *Proceedings of the 5th European Conference on Computer Systems*, ser. EuroSys '10. New York, NY, USA: ACM, 2010, pp. 265–278. [Online]. Available: http://doi.acm.org/10.1145/1755913.1755940

[23] M. Zaharia, A. Konwinski, A. D. Joseph, R. H. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments." in *OSDI*, vol. 8, no. 4, 2008, p. 7.

Authorized licensed use limited to: Auburn University. Downloaded on November 17,2026 at 23:05:27 UTC from IEEE Xplore. Restrictions apply.