Towards Energy Efficiency in Heterogeneous Hadoop Clusters by Adaptive Task Assignment

Dazhao Cheng*, Palden Lama†, Changjun Jiang‡ and Xiaobo Zhou*

*Department of Computer Science, University of Colorado, Colorado Springs, USA
†Department of Computer Science, University of Texas at San Antonio, San Antonio, USA
‡The Key Laboratory of Embedded System and Service Computing, Tongji University, China
Email addresses: dcheng@uccs.edu, palden.lama@utsa.edu, cjjiang@tongji.edu.cn, xzhou@uccs.edu

***Abstract*—The cost of powering servers, storage platforms and related cooling systems has become a major component of the operational costs in big data deployments. Hence, the design of energy-efficient Hadoop clusters has attracted significant research attentions in recent years. However, existing studies do not consider the impact of the complex interplay between workload and hardware heterogeneity on energy efficiency. In this paper, we find that heterogeneity-oblivious task assignment approaches are detrimental to both performance and energy efficiency of Hadoop clusters. Importantly, we make a counter-intuitive observation that even heterogeneity-aware techniques that focus on reducing job completion time do not necessarily guarantee energy efficiency. We propose a heterogeneity-aware task assignment approach, E-Ant, that aims to minimize the overall energy consumption in a heterogeneous Hadoop cluster without sacrificing job performance. It adaptively schedules heterogeneous workloads on energy-efficient machines, without a priori knowledge of the workload properties. Furthermore, it provides the flexibility to trade off energy efficiency and job fairness in a Hadoop cluster. E-Ant employs an ant colony optimization approach that generates task assignment solutions based on the feedback of each task's energy consumption reported by Hadoop TaskTrackers in an agile way. Experimental results on a heterogeneous cluster with varying hardware capabilities show that E-Ant improves the overall energy savings for a synthetic workload from Microsoft by 17% and 12% compared to Fair Scheduler and Tarazu, respectively.**

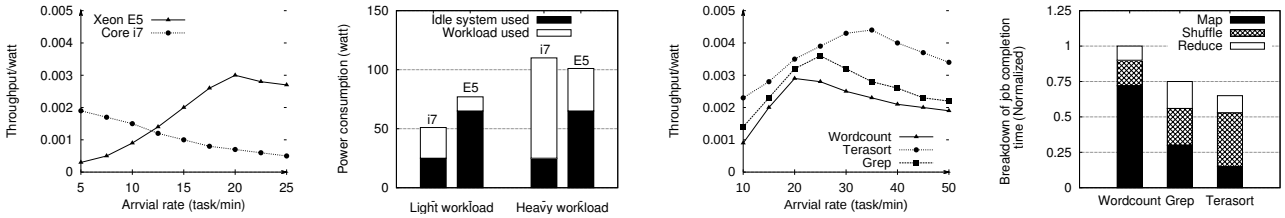
I. Introduction

As big data becomes a norm in various industries, the use of Hadoop framework to analyze the ever-increasing volume of data will keep growing [1], [2]. This trend is driving up the need for designing energy-efficient Hadoop clusters in order to reduce the operational costs and the carbon emission associated with its energy consumption [3], [4], [5]. However, prevalent power management techniques, which involve server consolidation and turning off idle machines, are not readily applicable to Hadoop clusters. It is due to the distributed data-storage and replication requirement of Hadoop platforms, which is essential for job performance and fault tolerance. Furthermore, a key challenge that has been often overlooked by existing studies is the complex interplay between the *heterogeneity* in hardware [6] and workload [7] characteristics, which is prevalent in production clusters.

A recent report [6] from Google suggests that there are more than ten generations of machines with different specifications in its production cluster. Such hardware heterogeneity occurs because servers are gradually upgraded and replaced in large scale clusters. Hardware heterogeneity could also arise from the possibility of deploying low power nodes in server clusters [8]. At the same time, production clusters are commonly shared by multiple users for diverse workloads with different resource demands, priorities and performance objectives. The co-location of heterogeneous workloads on the shared server infrastructure has been an important trend for gaining resource utilization efficiency.

In this paper, we observe that the heterogeneity of both machine and workload in Hadoop clusters causes complex and time-varying energy consumption characteristics at run-time. As shown by our motivational case study in Section II, the energy efficiency of heterogeneous machines varies with the type of workload as well as the rate at which the Hadoop tasks are assigned to the machines. Consequently, heterogeneity-oblivious approaches could result in both energy wastage and performance degradation. For example, we ran a Hadoop application from the PUMA benchmark, i.e., *Wordcount*, with 50 GB input data from Wikipedia, on a Core i7 based desktop and an Atom based server respectively. We found that the desktop takes 63 minutes of run time and 183 KJ of energy while Atom takes 178 minutes and 136 KJ to complete the job. It implies that job scheduling approaches that merely aims to reduce the job completion time do not guarantee a reduction in energy consumption of heterogeneous machines.

We propose a heterogeneity-aware task assignment approach, E-Ant, that aims to minimize the overall energy consumption in a heterogeneous Hadoop cluster without degrading job performance. There are several challenges in achieving the stated goals. First, a static task assignment policy based on workload and hardware profiling may not be effective due to the dynamic nature of energy usage characteristics, and uncertainties about the diverse workload-mix. Therefore, the task assignment policy needs to be adaptive. Second, the difficulty of directly measuring energy usage at the workload-level granularity introduces the need for an accurate energy usage model. Furthermore, the accuracy of energy usage estimation can be adversely affected by transient system noise attributed to data skew, network contention, etc. Third, the objective of achieving energy efficiency through adaptive task



(a) Heterogeneous platforms. (b) Power consumption. (c) Heterogeneous workloads. (d) Resource preference.

Fig. 1. Energy efficiency changes with platforms and workloads on a heterogeneous Hadoop cluster.

assignment may inadvertently conflict with job fairness.

We address the above challenges, and make the following contributions:

- We design an effective approach, E-Ant, to improve the energy efficiency of heterogeneous Hadoop clusters through adaptive task assignment. E-Ant's core is an ant colony optimization based algorithm that adaptively generates task assignment solutions based on the feedback of energy usage on heterogeneous machines. It monitors task execution of Hadoop jobs comprising of multiple waves of tasks, and optimizes the task assignments on-the-fly.
- To analyze the energy usage at workload-level granularity, we develop a model that estimates the energy consumption of completed tasks based on the time-varying CPU utilizations of its execution process and the amount of time that the tasks run at different CPU utilizations. This is motivated by the observation that CPU resource is the major power consumer in most clusters [9].
- To improve the robustness of our energy model against transient system noise, we develop a technique to estimate the energy usage of a Hadoop task by averaging the energy estimates of the same or similar job's tasks across homogeneous subset of machines from the entire cluster.
- We introduce a tuning parameter in the ant colony optimization problem to enable flexible tradeoff between energy efficiency and job fairness. Furthermore, we conduct sensitivity analysis of important tuning parameters that are used in E-Ant design.

We have implemented E-Ant in open-source Hadoop and performed comprehensive evaluations with representative Hadoop benchmark applications. Experimental results in a heterogeneous cluster with varying hardware capabilities show that E-Ant improves the overall energy consumption of a synthetic workload from Microsoft by 17% and 12% compared to Hadoop Fair Scheduler and Tarazu [10] respectively.

The rest of this paper is organized as follows. Section II presents a motivation example. Section III describes the system design. Section IV presents the online algorithm. Section V gives the testbed implementation. Section VI presents the experimental results and analysis. Section VII reviews related work. Section VIII concludes the paper.

II. Motivation

We conducted a case study on a heterogeneous Hadoop cluster composed of two types of machines listed in Table I. Three

TABLE I
MULTIPLE MACHINE TYPES IN THE CLUSTER.

Machine model	CPU	Memory	Disk
Desktop Core i7	8*3.4GHz	16 GB	1 TB
PowerEdge Xeon E5	24*1.9GHz	32 GB	1 TB

MapReduce applications from the PUMA benchmark [11], i.e., *Wordcount*, *Terasort* and *Grep*, each with 300 GB input data, were run on the cluster. We measured the power consumption and the task throughput on the different type machines under the various task submission rates. We observed the energy efficiency of individual machines in terms of the throughput per watt, which is widely used by previous studies [12]. We compared the energy efficiency in two different scenarios – homogeneous workload on heterogeneous hardware and heterogeneous workload on homogeneous hardware.

Heterogeneous hardware impact. Figure 1(a) shows that the energy efficiency of machines with different hardware configurations varies differently as we change the task arrival rate. The Core-based desktop achieved better energy efficiency when the task arrival rate was lower than 12 tasks/min. However, the energy efficiency on the Xeon-based server was better when the task arrival rate was higher than 12 tasks/min. In order to analyze the root cause of such behavior, we measured the power consumption of the two machines under two representative scenarios in the experiment: light workload (10 tasks/min) and heavy workload (20 tasks/min). Figure 1(b) shows that most power consumption of the Xeon based server comes from the idle system usage when the server utilization is relatively low. However, when the utilization increases, the power consumption of Xeon server increases slowly compared with that of the core based desktop. This is the reason that Xeon-based server is more energy-efficient for heavy workload while less energy-efficient for light workload. Such characteristic of heterogeneous hardware motivates us to improve the overall energy efficiency of the cluster by dynamically assigning tasks to different machines.

Heterogeneous workload impact. Figure 1(c) shows the variation in the energy efficiency of a homogeneous cluster consisting of Xeon-based servers only, when different Hadoop benchmarks were executed. The result suggests that *Wordcount*, *Grep*, and *Terasort* benchmarks achieved their maximum energy efficiency at different task arrival rates,

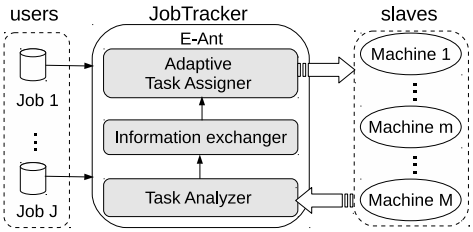


Fig. 2. The system framework of E-Ant.

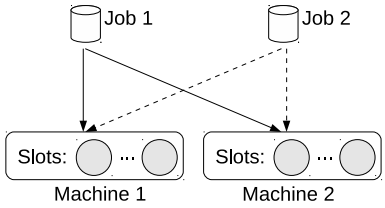


Fig. 3. Task assignment by applying ACO.

which were 20, 25, and 35 tasks/min, respectively. This is due to the fact that different workloads have various resource demands, which in turn impact the energy usage pattern. Figure 1(d) shows the breakdown of job completion time of the three applications used in the experiment. It reveals that *Wordcount* is map-intensive (i.e., CPU-intensive) while *Terasort* and *Grep* are shuffle/reduce-intensive (i.e., IO-intensive). The result demonstrates the importance of considering the workload heterogeneity when assigning tasks in order to maximize energy efficiency.

[**Summary**] The above study shows that the energy efficiency of a Hadoop cluster can be substantially improved from two aspects:

- Selecting energy-efficient hosting machines based on the different power characteristics of machines;
- Assigning suitable number of tasks based on the different resource demand of workloads.

In other words, Hadoop tasks should be assigned to the machines that deliver the highest energy efficiency for the specific workload type. However, the dynamic nature of energy usage characteristics and uncertainties about the workload mix in production environments further complicate the problem. These observations motivate us to develop a self-adaptive and heterogeneity-aware task assignment approach to improve the energy efficiency of a Hadoop cluster.

III. E-Ant Design

A. Architecture

E-Ant is a self-adaptive task assignment approach that aims to improve the energy efficiency of a heterogeneous Hadoop cluster. As shown in Figure 2, E-Ant fits seamlessly with the Hadoop architecture. Its main components are:

- **Adaptive task assigner** uses an Ant Colony Optimization (ACO) approach to make task assignment decisions based on the task-level feedback reported by the *task analyzer*. The basic idea is that the Hadoop slave machines that are highly ranked by the task analyzer will likely be assigned with more of the same type of tasks.
- **Task analyzer** uses energy models to estimate the energy consumption of individual tasks at different machines. Each model captures the energy usage characteristics of a particular machine type, and takes into account the CPU utilization of individual tasks, and its execution time to estimate task-level energy usage.
- **Information exchanger** facilitates the grouping of task information related to homogeneous subset of machines and jobs in a heterogeneous Hadoop cluster. E-Ant uses this information to improve the estimation accuracy of its energy models in the presence of system noise.

E-Ant is designed to take advantage of the fact that Hadoop applications running in a shared cluster often execute several waves of tasks for big data. It operates as follows. When jobs are submitted to the `JobTracker`, it initially follows Hadoop's default behavior to assign the tasks to various `TaskTrackers`. At the end of each control interval, the task analyzer in the `JobTracker` estimates the energy consumption of the completed Hadoop tasks, and ranks the machines accordingly. The adaptive task assigner uses the rankings to adjust its task assignment policy for the next control interval. The `JobTracker` schedule all tasks according to the new task assignment policy within the next control interval. After rounds of adjustment, task assignments of various Hadoop jobs converge to energy-efficient solutions. This process is repeated until the job completes after several waves of task execution. Thus, given the obtained energy model, E-Ant does not require a priori knowledge of submitted jobs due to its self adaptation.

B. Applying ACO to Task Assignment Problem

We now present a high-level overview of how ACO can be used to solve the dynamic task assignment problem in a heterogeneous Hadoop cluster. ACO is inspired by a colony of ants that work together to find the shortest path between their nest and food source. It is a probabilistic technique for solving computational problems which can be reduced to finding good paths through graphs. Here, an ant is a simple computational agent, which iteratively constructs a solution for the problem at hand.

Figure 3 illustrates the problem of assigning tasks belonging to multiple Hadoop jobs to different machines. We consider each job as an ant colony and each task as an ant. The assignment of a task to a machine is treated as a path that an ant can take. In the rest of this paper, job and colony, task and ant, thus are used interchangeably. The goodness of a path is evaluated based on the energy consumption of a task completed on a particular machine. This information is encoded as the pheromone value of a path, and is updated as more tasks are completed. E-Ant periodically updates its task assignment policy based on the pheromone values of various paths at that time. Thus, E-Ant applies the ACO approach to adaptively choose energy-efficient hosts for task assignment.

TABLE II
CONSTRUCTION GRAPH OF 3 MACHINES AND 4 TASKS.

$HM \setminus TS$	T_1^1	T_2^1	T_1^2	T_2^2
M_1	$E(T_1^1(1))$	$E(T_2^1(1))$	$E(T_1^2(1))$	$E(T_2^2(1))$
M_2	$E(T_1^1(2))$	$E(T_2^1(2))$	$E(T_1^2(2))$	$E(T_2^2(2))$
M_3	$E(T_1^1(3))$	$E(T_2^1(3))$	$E(T_1^2(3))$	$E(T_2^2(3)$

IV. ADAPTIVE TASK ASSIGNMENT

In this section, we present the detailed design of ACO-based algorithm for adaptive task assignment. First, we formulate the dynamic task assignment problem in a heterogeneous environment. Then we present a model to evaluate the energy consumption of each task based on its CPU utilization and task execution time. Finally, we present the ACO-based optimization algorithm to obtain an adaptive task assignment scheme, and describe the related design components.

A. Problem Statement and Graph Construction

1) Problem Formulation: We consider a Hadoop cluster with M slave machines, indexed 1,... , m ($m \leq M$). Assume that there are J jobs hosted in the cluster and each job (e.g., job j) consists of a number of tasks, i.e., $T_1^j, ..., T_n^j$ ($n \leq N$). We formalize E-Ant's goal as assigning the tasks of Hadoop jobs to different machines in order to minimize the overall energy consumption under certain constraints. The problem can be expressed as follows:

$$\begin{aligned} \text{Min} \quad & \sum_{j=1}^{J}\sum_{n=1}^{N}\sum_{m=1}^{M}[E(T_n^j(m))], \\ \text{s. t.} \quad & \sum_{j=1}^{J}\sum_{n=1}^{N}|T_n^j(m)| \leq m_{slot}. \\ \text{s. t.} \quad & P(j,m) = f(H) \end{aligned} \tag{1}$$

Here, $E(T_n^j(m))$ represents the energy consumption of the n_{th} task from the j_{th} job that is assigned to the m_{th} machine. The first constraint ensures that the number of concurrent task executions from all jobs on the machine m must be bounded by the number of available slots on that machine. The second constraint ensures that the probability $P(j,m)$ of assigning a task from the j_{th} job to the m_{th} machine is determined by a heuristic function based on data locality and job fairness.

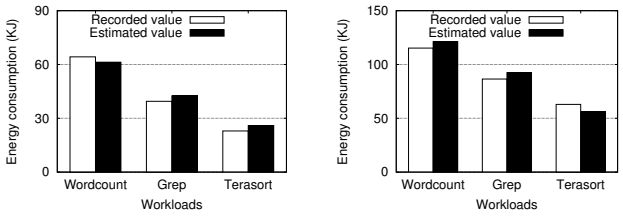
2) Construction Graph: Table II illustrates a visual representation of the task assignment problem formulated in Equation 1. We draw an analogy between the task assignment problem and a colony of ants exploring alternative paths towards its destination. Given a table of m rows and n columns, an ant seeks to travel in such a way that all of the following constraints are satisfied: (1) one and only one cell is visited for each of the columns; (2) the number of the visited cells on the same row is no greater than the available slots on individual machines (i.e., satisfy the constraint in Eq. 1).

B. Energy Consumption Model

In order to make task assignment decisions that improve energy efficiency, it is necessary to evaluate the energy used by each task execution. E-Ant uses a simple and feasible energy model based on CPU power consumption to evaluate the energy consumption of tasks at a fine-grained level. Since a Hadoop task is executed on a JVM hosted at a slave machine, we consider the energy consumed by this JVM as the task energy consumption. We estimate the energy consumption of an individual task (JVM) based on its CPU resource utilization at process-level. The energy consumed by the n_{th} task from the j_{th} job on machine m is estimated as follow:

$$E(T_n^j(m)) = \sum_{T_{start}}^{T_{finish}}(\frac{Power_m^{idle}}{m_{slot}} + \alpha_m \times u(T_n^j(m))) \times \Delta t, \tag{2}$$

Here, $Power_m^{idle} \in \mathbb{R}^+$ is the power consumption of machine m when it is idle. It is divided by the number of slots available in the machine, since each Hadoop task occupies one slot. $\alpha_m \in \mathbb{R}^+$ represents how the power consumption changes with the change in CPU utilization. It can be obtained by using a standard system identification technique, the least squares method. Both metrics are constant for a particular machine type. $u(T_n^j(m))$ is the average CPU utilization of the execution process for the task $T_n^j(m)$ during the time interval Δt. T_{start} and T_{finish} are the task start time and the task finish time. These metrics are collected and reported by the `TaskTrackers` running on each machine.



(a) Desktop Core i7. (b) Poweredge Xeon E5 server.

Fig. 4. Estimation accuracy of energy model in heterogeneous machines.

In order to evaluate the accuracy of the model, we compared the actual energy consumption of a machine running a particular job with the sum of the estimated energy consumption of all tasks running on that machine. We set the granularity of CPU utilization measurement Δt to three seconds (the default heartbeat interval of Hadoop). Figure 4(a) and 4(b) show the actual and estimated energy consumption of a Dell desktop, and a Xeon E5 server when running the *Wordcount*, *Grep*, and *Terasort* benchmarks respectively. We evaluated the estimation accuracy of the model in terms of the normalized root mean square error (NRMSE), a standard metric for deviation. Our results show that the measured and estimated data are very close, with the NRMSE smaller than 8%.

C. ACO-based Adaptive Task Assignment

1) Solution Construction: Given a set of tasks and heterogeneous machines, E-Ant follows the principle of ant colony optimization to stochastically assign the tasks to the available machines. The probability that E-Ant will assign a task of the j_{th} job to the m_{th} machine during the time interval $t+1$ is

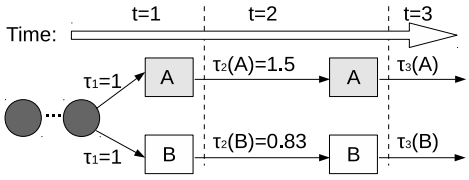


Fig. 5. The probabilistic decision-making process example.

given by:

$$P_{t+1}(j,m) = \frac{\tau_t(j,m)}{\Sigma_{m' \in M} \tau_t(j,m')}. \quad (3)$$

Here, $\tau_t(j,m)$ is the pheromone value associated with a particular path for a colony of ants. In other words, it indicates the goodness of assigning a task of the j_{th} job on the m_{th} machine. $\Sigma'_m \in M$ denotes the set of all completed tasks of the j_{th} job across multiple machines.

Figure 5 shows a simple example to introduce the decision-making process of the proposed approach for a single job (ant colony). Here, we omit the symbol j from $\tau_t(j,m)$ for better readability. Assume that a set of tasks need to be assigned to two machines, i.e., machine A and machine B. At the first time interval (t=1), there is an equal pheromone ($\tau_1(A) = \tau_1(B) = 1$) for the two possible task assignment paths. Correspondingly, tasks have the same probability (i.e., $P_1(A) = P_1(B) = \frac{1}{1+1} = 50\%$) to be assigned to the machine A and B. However, the pheromone of machine A ($\tau_2(A) = 1.5$) is larger than that of machine B ($\tau_2(B) = 0.83$) during the second interval. It means that machine A has a higher probability ($P_2(A) = \frac{1.5}{1.5+0.83} = 64\%$) to host such type of tasks than machine B does ($P_2(B) = \frac{0.83}{1.5+0.83} = 36\%$). In the next section, we describe the merits and mechanism of updating the pheromone values.

2) Pheromone Updating: E-Ant follows two principles of ant colony optimization: Randomness and Positive Feedback.

- **Randomness** is incorporated in E-Ant by the fact that its task assignment policy is based on probability as shown in Equation 3. This allows E-Ant to explore new solutions which may lead to a globally optimal solution, instead of getting stuck in a local optimum.
- **Positive feedback** exploits good solutions by laying more pheromone on the task assignment paths that lead to less energy consumption.

E-Ant updates the pheromone value of various assignment paths periodically based on the task-level feedback collected from the previous control interval as follows:

$$\tau_{t+1}(j,m) = (1-\rho)\tau_t(j,m) + \rho \Sigma_{n \in N} \Delta\tau_t^n(j,m) \quad (4)$$

$$\Delta\tau_t^n(j,m) = \frac{\Sigma_{m' \in M} \Sigma_{n' \in N} E(T_{n'}^j(m'))/N}{E(T_n^j(m))}. \quad (5)$$

Here, $\Delta\tau_t^n(j,m)$ represents the energy efficiency of completing the n_{th} task of the j_{th} job on the m_{th} machine, and is responsible for an increase in the pheromone value

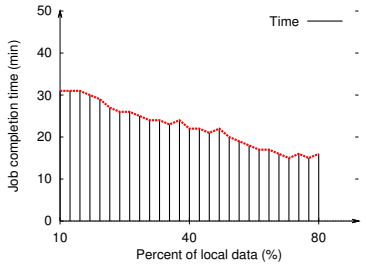


Fig. 6. Impact of data locality (*Wordcount* benchmark).

of the assignment path. It is calculated as the ratio of the average energy consumption of all completed tasks of the j_{th} job to the energy consumption of its n_{th} task on the m_{th} machine. ρ is the pheromone evaporation coefficient. It is used to avoid unlimited accumulation of pheromone trails from previous control intervals. The proposed pheromone updating mechanism encourages a balanced form of exploitation of previous experiences (the first item) and exploration of new or improved paths (the second item). Equation 4 also implies that the higher the task completion rate of the j_{th} job on the m_{th} machine, the greater the chance of updating the pheromone value of that path. Thus, the task assignment policy favors those machines that do more work and consume less energy.

Continuing with the example in Figure 5, we now assume that the energy consumption of individual tasks on machine A is 2 KJ and that on machine B is 3 KJ during the first interval. Suppose that machine A completes two tasks, and machine B completes one task during the interval. The evaporation coefficient ρ is set to 0.5. Then we obtain the value of pheromone on machine A as: $\tau_2(A) = (1-0.5)*1+0.5*2*\frac{(2+2+3)/3}{2} = 1.66$ and $\tau_2(B) = (1-0.5)*1+0.5*\frac{(2+2+3)/3}{3} = 0.88$. This means that E-Ant has a higher probability of assigning tasks to machine A, which is more energy efficient than machine B.

3) Multi-job scenario: E-Ant allows multiple ant colonies to construct their solutions in parallel for multiple jobs. Each ant colony updates its pheromone values periodically as described in the single job scenario. Furthermore, E-Ant uses pheromone updating as an interaction mechanism not only between the tasks of the same job but also among multiple jobs. For this purpose, E-Ant applies negative feedback to update the pheromone value across jobs as follows,

$$\Delta\tau_t^n(j',m) = \begin{cases} +\Delta\tau_t^n(j,m) & \text{if j = j'}, \\ -\Delta\tau_t^n(j,m) & \text{Otherwise.} \end{cases} \quad (6)$$

The negative feedback mechanism applies an aggressive pheromone updating approach among competing jobs. It encourages the most energy-efficient machine to host more tasks from a specific job by giving a positive pheromone update. At the same time, it gives a negative pheromone update $-\Delta\tau_t(m)$ for other jobs to reduce their probability of assigning tasks on this machine.

4) Heuristic Information: We present a heuristic function that further improves the ACO-based adaptive task assignment by incorporating two factors: *data locality* and *job fairness*.

Prioritizing data locality. The first property of the heuristic function is designed to prioritize data locality when making task assignment decisions. This is because Hadoop job execution times can be substantially reduced by assigning tasks to the machine that can access the input data locally. The motivation is that better data locality will result in shorter job completion times, and potentially lead to better energy efficiency. Figure 6 depicts the job completion times of multiple *Wordcount* jobs with the same size input data but different data locality. It demonstrates that job completion time decreases as the input data locality increases.

Providing job fairness. The second property of the heuristic function is designed to promote fair share of resources among competing jobs in a shared Hadoop cluster. Without this property, it is possible that only certain type of jobs make more progress than others. For example, since short jobs may finish more tasks within a control interval than long jobs, they get more chances to update the pheromone values of their task assignment paths, and give negative pheromone update to competing jobs. As a result, the task assignment policy may unfairly lead to starvation of longer jobs.

Heuristic function. The heuristic function is given by:

$$\eta_{t+1}(j) = \begin{cases} \infty & \text{if task has local data,} \\ \frac{1}{1-\frac{S^j_{min}-S^j_{occ}}{S_{pool}}} & \text{Otherwise.} \end{cases} \quad (7)$$

where S^j_{min} is the minimum share of the j_{th} job in terms of the number of Hadoop slots. S^j_{occ} is the number of slots occupied by the j_{th} job. S_{pool} is the minimum share of the Hadoop user who is running the job. Similar to the Hadoop Fair Scheduler, $\Sigma_{j \in J} S^j_{min} = S_{pool}$ holds true. For the sake of clarity, we consider a single user system. Hence, S_{pool} is equal to the total number of slots available in the cluster. E-Ant incorporates the above heuristic information into its task assignment policy by updating Equation 3. The probability that E-Ant will assign a task of the j_{th} job to the m_{th} machine is now given by:

$$P_{t+1}(j,m) = \frac{\tau_t(j,m)[\eta_{t+1}(j)]^\beta}{\Sigma_{m' \in M} \tau_t(j,m')}. \quad (8)$$

where β is a control knob that determines the relative influence of heuristic and pheromone information on the task assignment decisions.

The first term in Equation 7 indicates that the task assignment policy should give the highest priority to data local tasks if available (i.e.,$\eta_{t+1}(j) = \infty$). The second term indicates the degree of unfairness experienced by the j_{th} job. The higher the degreee of unfairness, the greater the need to schedule the tasks belonging to this job. From Equations 7 and 8, we can conclude the following: If $S_{min} = S_{occ}$, the value of $\eta_{t+1}(j)$ will be one. In other words, the j_{th} job already has its fair share of resources. Hence, the fairness factor will not have any impact on the probability of assigning the j_{th} job to the m_{th} machine. If $S_{occ} < S_{min}$, the value of $\eta_{t+1}(j)$ will be greater than one. Furthermore, the larger the difference between S_{occ} and S_{min}, the greater the value of $\eta_{t+1}(j)$. It means that the job is experiencing unfairness, and hence the probability of assigning the j_{th} job to the m_{th} machine will be increased. If $S_{occ} > S_{min}$, the value of $\eta_{t+1}(j)$ will be a positive number smaller than one. The larger the difference, the smaller the value of $\eta_{t+1}(j)$. It means that the job is using more than its fair share of resources, and hence the probability of assigning the j_{th} job to the m_{th} machine will be reduced.

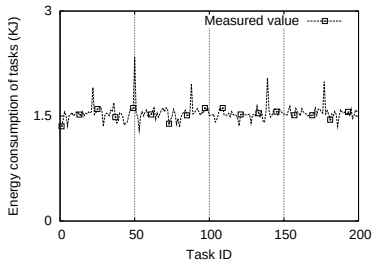


Fig. 7. Impact of system noise (*Wordcount* benchmark on T420 machine).

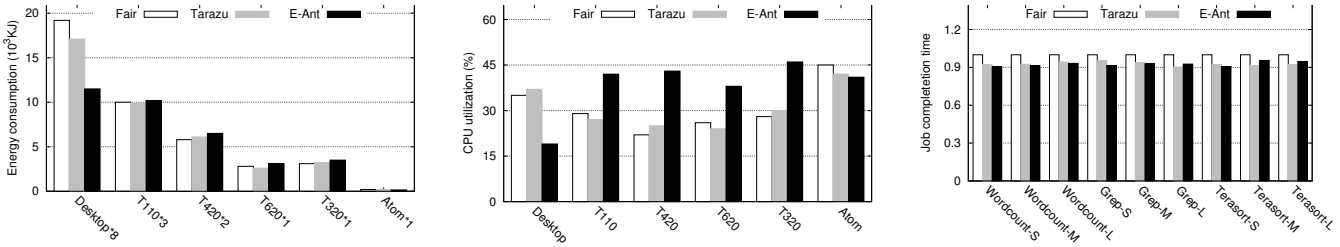
D. Exchange Strategies for Robustness Against System Noise

We observe that the presence of system noise in a Hadoop cluster poses a challenge in accurately evaluating the energy efficiency of a particular type of machine for certain jobs. We define system noise as the transient and anomalous behavior of certain tasks of a given job, which may be attributed to multiple factors such as data skew, network congestion, etc. Such system noise manifests itself as fluctuation in CPU utilization, and straggling tasks that run slower than their expected speed on a particular machine type. As a result, the energy usage estimated by the model in Equation 2 may not reflect the true energy efficiency of a machine. Figure 7 shows the impact of system noise on the estimated energy consumption of individual tasks while running a *wordcount* job on a T420 machine.

We address this challenge by leveraging the presence of homogeneous subset of machines and jobs in a heterogeneous Hadoop cluster. Since homogeneous machines are supposed to deliver similar energy efficiency for the tasks of the same or similar jobs, the evaluation of energy efficiency will be more accurate if we consider the energy usage estimates of these groups together. E-Ant utilizes two levels of information exchange, i.e., machine-level exchange and job-level exchange.

Machine-level exchange. E-Ant divides the heterogeneous Hadoop cluster into a number of homogeneous sub-clusters based on their hardware configurations. The hardware information is collected by the `JobTracker` on the master node using the heartbeat signal from the `TaskTrackers`. The pheromone update value used in Equation 4 is now defined as $\Delta\tau^n_t(j,m) = Avg_{m' \in M_h} \Delta\tau_t(j,m')$ where M_h is a set of homogeneous machines. Hence, the goodness of a task assignment path for a particular job is determined by the average available experiences of the completed tasks that visited those homogeneous machines in the previous interval.

Job-level exchange. E-Ant divides the heterogeneous workload-mix into a number of homogeneous jobs based on



(a) Energy consumption. (b) CPU utilization. (c) Job completion time.

Fig. 8. Comparing E-Ant with Fair Scheduler and Tarazu.

TABLE III
MSD WORKLOAD CHARACTERISTICS.

Size	% Jobs	Input size	# Maps	# Reduces
Small	40%	1-100GB	16-1600	4-128
Medium	20%	0.1-1TB	1600-16000	128-256
Large	10%	1-10TB	16000-160000	256-1024

their resource demands. The job-level information exchange is achieved by using an average evaluation function to evaluate the pheromones of homogeneous jobs. It aims to share the knowledge from all homogeneous jobs executed on the homogeneous machines. More specifically, the pheromone update value is defined as $\Delta\tau_t^n(j,m) = Avg_{m'\in M_h}^{j'\in J_h}\Delta\tau_t^n(j',m')$, where J_h is a set of homogeneous jobs.

V. IMPLEMENTATION

A. Hadoop Modification

We implemented E-Ant by modifying the Java classes `JobTracker`, `TaskTracker` and `TaskReport` based on Hadoop version 1.2.1. We added a new interface `taskEner`, which is used to estimate the energy consumption of individual tasks while comparing them hosted in different slave nodes. Each record of task-level energy consumption is tagged with its corresponding `AttemptTaskID`. Additionally, we added another new interface `Optimizer` to implement the ACO optimization. During job execution, we created a method `taskAnalyzer` to collect the status of each completed task by using `TaskCounter` and `TaskReport`. The CPU utilizations and the execution times of tasks are reported by `TaskTrackers` via heartbeat connection periodically.

B. Experiment Setup

We evaluate E-Ant on a physical cluster composed of 1 Atom (4-core CPUs, 8 GB RAM and 1 TB hard disk), 3 T110 (8-core CPUs, 16 GB RAM and 1 TB hard disk), 2 T420 (24-core CPUs, 32 GB RAM and 1 TB hard disk), 1 T320 (12-core CPUs, 24 GB RAM and 1 TB hard disk), 1 T620 (24-core CPUs, 16 GB RAM and 1 TB hard disk), and 8 Dell desktops (8-core CPUs, 16 GB RAM and 1 TB hard disk). The master node is hosted on one Dell desktop in the cluster. The servers are connected with Gigabit Ethernet. Each slave node is configured with four map slots and two reduce slots. The block size of HDFS is set to 64MB in the experiment. E-Ant is based on the version 1.2.1 of Hadoop implementation. We measure the energy consumption of each machine directly using the WattsUP Pro power meter [13]. We implement the proposed adaptive assignment algorithm on the $Master$ node of the cluster. It dynamically generates new scheme to adjust the task assignment in each control interval (i.e., 5 minutes). The selection of the control interval is a trade-off between the solution searching speed and average task execution time.

C. Real-world Workloads

To understand the effectiveness of E-Ant in a production environment, we used a synthetic workload, "MicroSoft-Derived (MSD)", which models the production workload of 174,000 jobs in Microsoft datacenter in a single month in 2011 [14]. MSD mimics the distribution characteristics of the production jobs by running *Wordcount*, *Terasort* and *Grep* applications from the PUMA benchmark [11] with various input data sizes. It is a scaled-down version of the workload studied in [14] since our cluster is significantly smaller. We scale down the workload in two ways: we reduce the overall number of jobs to 87, and eliminate the largest 10% of jobs and the smallest 20% of jobs.

The Hadoop implementation comes with several configuration parameters that need to tuned for job performance. As shown in Table IV, we set the Hadoop configurations according to the rules recommended by Cloudera [15]. We used the same configurations for evaluating Fair Scheduler, Tarazu, and E-ANT.

TABLE IV
HADOOP PARAMETER CONFIGURATION.

Parameter	*Wordcount*	*Grep*	*Terasort*
io.sort.factor	10	10	10
io.sort.mb	100mb	200mb	150mb
io.sort.record.percent	0.35	0.25	0.15
io.sort.spill.percent	0.8	0.6	0.75
io.file.buffer.size	4k	16k	32k
mapred.child.java.opts	200	300	250

VI. EVALUATION

We compare the performance of E-Ant with two competitors: Fair Scheduler, a representative approach that is used to

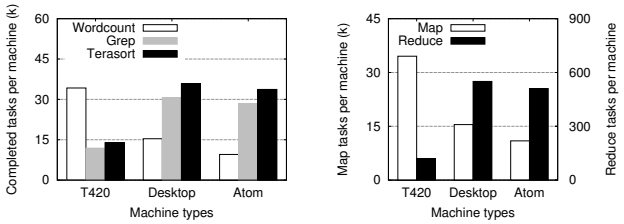
share a Hadoop cluster among multiple jobs, and Tarazu [10], a communication-aware load balancing scheduler for improving heterogeneous MapReduce performance. First, we evaluate the effectiveness of E-Ant in improving the energy efficiency of a heterogeneous Hadoop cluster. Second, we analyze the adaptiveness of E-Ant task assignment policy. Third, we evaluate the effectiveness of the exchange strategy. Finally, we present the sensitivity analysis of E-Ant design parameters.

A. *Effectiveness of E-Ant*

Reducing energy consumption. Figure 8(a) shows the impact of Fair Scheduler, Tarazu, and E-Ant on the overall energy consumptions of six different machine types used in our experiments. The results demonstrate that E-Ant achieves significant energy savings on the eight Desktop machines, while using slightly more energy on the other types of machines. Note that the energy saving would be achieved on other machines if there are no Desktop machines in the cluster. E-Ant improves the overall energy consumption of MSD workload by 17% and 12% compared with Fair Scheduler and Tarazu, respectively. This is due to the fact that E-Ant distributes the Hadoop tasks in heterogeneity-aware manner to improve the overall energy efficiency. On the other hand, Fair Scheduler is heterogeneity-oblivious and Tarazu aims to improve application performance rather than energy efficiency. The results also reveal that Tarazu is more energy efficient than Fair Scheduler since Tarazu could reduce job execution time by communication-aware load balancing.

Impact on CPU utilization. Figure 8(b) shows the CPU utilizations of the various machine types that resulted from different scheduling approaches. It illustrates that T420 server has only 20% utilization by using Fair Scheduler and Tarazu while it has 40% utilization by using A-Ant. In contrast, E-Ant achieves a lower utilization on the desktop compared to Fair Scheduler and Tarazu do. This is due to E-Ant's heterogeneity-aware task assignment, which favors T420 server over the desktop in order to improve the overall energy efficiency. As discussed in Section II, the Xeon-based server is more energy efficient for heavy workload while the Core-based desktop is more efficient for light workload.

Impact on job completion time. Figure 8(c) compares the impact of Fair Scheduler, Tarazu and E-Ant, respectively on Hadoop job performance. Here, the job completion time is normalized with respect to the job completion time achieved by the Fair Scheduler. The result shows that both Tarazu and E-Ant improve the job completion time compared to Fair Scheduler. Tarazu achieves performance gain by avoiding bursty network traffic during map computation. On the other hand, E-Ant improves the job performance due to its heterogeneity-aware task assignment. It favors those machines that are able to finish more tasks within a control interval, since the pheromone values associated with their assignment paths are updated more frequently. In a few cases, the performance improvement by E-Ant is slightly less than that of Tarazu. This is due to the fact that E-Ant may allow some slow task executions, in order to achieve significant energy savings.



(a) Varying workload assignment. (b) Map and Reduce assignment.

Fig. 9. Task assignment adaptiveness by workloads and tasks.

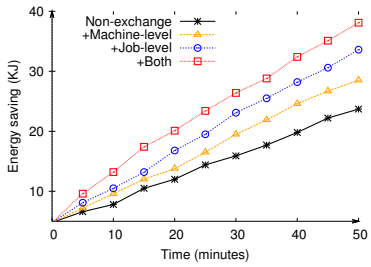


Fig. 10. Effectiveness of the information exchange strategy.

Overall, E-Ant achieves consistently better energy efficiency than Fair Scheduler and Tarazu do.

B. *Adaptive Task Assignment*

We now demonstrate how E-Ant's task assignment policy adapts to different Hadoop workloads and tasks types.

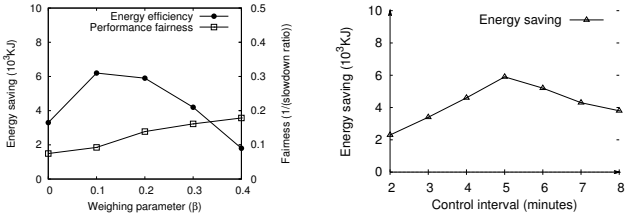
Adaptiveness by workload type. Figure 9(a) shows E-Ant task assignment distribution on three representative machines types for different workloads. The results show that T420 server hosts more tasks of the *Wordcount* job while the desktop and Atom host more tasks of the *Grep* and *Terasort* jobs. This is due to the fact that compute-optimized machines such as T420 are more energy-efficient for CPU bound workloads (e.g., *Wordcount*), and the less powerful machines such as desktop and Atom are relatively more energy-efficient for I/O bound workloads (e.g., *Grep* and *Terasort*).

Adaptiveness by task type. Figure 9(b) shows E-Ant's task assignment distribution on three representative machines types for different task types. The results show that T420 server hosts more map tasks, while the desktop and Atom host more reduce tasks. This is due to the fact that compute-optimized machines (e.g., T420) are more energy-efficient for CPU intensive tasks, and the less powerful machines (e.g., desktop and Atom) are relatively more energy-efficient for I/O intensive tasks. The map tasks happen to be relatively more CPU intensive than the reduce tasks for our workload mix.

C. *Effectiveness of Information Exchange Strategy*

We study the impact of information exchange strategy on the amount of energy savings, and the search speed of E-Ant.

Impact on Energy Savings. As shown in Figure 10, we measure the energy savings achieved by E-Ant with different information exchange strategies over the default heterogeneity-agnostic Hadoop. The measurements are done at different



(a) Weighting parameter setting. (b) Control Interval setting.

Fig. 11. Sensitivity analysis of E-Ant design parameters.

times intervals as the jobs progress towards completion. As time progresses, more and more tasks are assigned to the heterogeneous machines. As a result, the energy savings achieved by E-Ant increases with time. The result shows that the machine-level exchange and job-level exchange strategies improve the energy savings by 7% and 10% respectively as compared with the no-exchange strategy. Figure 10 also illustrates that applying the two strategies together improves the energy savings by 15% compared with applying the no-exchange strategy. The improvement in energy savings compared to the no-exchange strategy is due to the fact that the exchange strategy enables E-Ant to make more accurate judgment about the energy efficiency of the machines in the presence of system noise.

Impact on Search Speed. The search speed of E-Ant is evaluated by measuring the time required to find a stable task assignment solution for a job running on the Hadoop cluster. We define a stable solution as the task assignment that has more than 80% tasks revisiting the same machines compared with the assignment in the previous interval [16]. A slow search speed will reduce the benefits of an energy-efficient task assignment, especially for small jobs. This is due to the fact that a small job may complete before E-Ant can find out energy-efficient hosting machines for this job. Our experiments reveal that the search speed of E-Ant is affected by the number of homogeneous machines in a heterogeneous cluster and the number of homogeneous jobs available for applying the exchange strategy.

D. Sensitivity Analysis of E-Ant design parameters

Weighting parameter impact. We measure the energy saving achieved by E-Ant over the default heterogeneity-agnostic Hadoop, and evaluate its job fairness for various values of the weighting parameter β in Equation 8. In order to measure job fairness, we compare the performance slowdown of the various Hadoop jobs. Slowdown is the normalized execution time, which is defined as the ratio between a job's actual execution time and its standalone execution time (the time when a job is running in the system alone) [17]. In a fair system, all jobs are expected to experience similar slowdown. Hence, we define job fairness as the inverse of the variance in performance slowdown of the submitted jobs. As shown in Figure 11(a), job fairness increases with the increasing value of β. This is due to the fact that E-Ant's task assignment policy gives fairness a higher priority than energy saving as the value of β increases. This also explains the decrease in energy saving when the value of β increases from 0.1 to 0.4.

However, there is an increase in energy saving when the value of β increases from zero to 0.1. The reason is that when the value of β is zero, E-Ant does not consider data locality in making task assignment decisions. As a result, a large number of non-local tasks may increase the job execution time, which in turn leads to more energy usage. On the other hand, when β has a nonzero value E-Ant gives the highest priority to data locality whenever local task execution is possible. Hence, there is an initial increase in energy savings as shown in Figure 11(a). However, as the value of β gets larger, the impact of favoring job fairness over energy saving becomes more prominent. Hence, the energy saving decreases.

Overhead and Control Interval Selection. E-Ant relies on Hadoop's daemon processes such as `JobTracker` and `TaskTrackers` to monitor task execution and perform adaptive task assignment. Hence, there is no overhead of launching any new process for E-Ant. It collects the energy consumption information of individual tasks from `TaskTrackers`, which is along with `TaskReport` by Hadoop default RPC connections. Furthermore, the self-adaptive ACO algorithm itself takes approximately 120 ms to complete. This overhead is negligible compared to E-Ant's control interval of 5 minutes.

We now analyze the impact of control interval selection on the energy saving achieved by E-Ant over default Hadoop. As shown in Figure 11(b), the energy saving first increases with increasing value of the control interval. However, the energy saving decreases for control intervals that are greater than 5 minutes. This is due to the fact that when the control interval is too short, E-Ant's adaptive task assigner does not have enough data samples to make an accurate judgment about the energy efficiency of heterogeneous machines. Hence, increasing the control interval has a positive impact on energy saving. However, when the control interval is too long, task assignments become less frequent which in turn reduces the possibility of energy saving.

VII. Related Work

Task scheduling: Many prior studies have shown that MapReduce performance can be significantly improved by various scheduling techniques [18], [19], [20]. The default FIFO Scheduler in Hadoop may not work well since a long job can exclusively take the computing resource on the cluster, and cause large delays for other jobs. Thus, many schedulers, e.g., Capacity Scheduler, Fair Scheduler, can share resources among multiple jobs. Wolf *et al.* described FLEX [19], a flexible allocation scheme for MapReduce workloads. Cho *et al.* designed Natjam [18], a system that provides support for prioritized MapReduce scheduling of the jobs with various deadlines. Our work differs from these efforts in that we investigate adaptive task assignment techniques to improve energy efficiency of heterogeneous Hadoop clusters.

Heterogeneous cluster: As heterogeneous hardware is prevalent in production environments, studies [10], [12], [21],

[22], [23] aim to improve MapReduce performance in heterogeneous clusters. Ahmad *et al.* [10] identified key reasons for MapReduce poor performance on heterogeneous clusters. Accordingly, they proposed an optimization based approach, Tarazu, to improve MapReuce performance by communication-aware load balancing. Zaharia *et al.* [21] designed a robust MapReduce scheduling algorithm, LATE, to improve the completion time of MapReuce jobs in a heterogeneous environment. Although these approaches achieved performance improvement, they paid little attention to optimizing energy efficiency for heterogeneous Hadoop clusters.

Energy efficiency: There are growing interests on energy-efficient MapReduce design with various techniques, e.g., cluster consolidating [24], delaying batch jobs [25] and load distribution [26]. Leverich *et al.* proposed covering subset scheme keeps one replica of every block within a small subset of machines called the covering subset. Lang *et al.* proposed AIS [27], an alternative energy management framework to reduce the energy consumption of Hadoop cluster. It dynamically controls the available nodes in the cluster to improve the system resource utilization. Chen *et al.* proposed BEEMR [24], an energy efficient MapReduce workload manager. Its key insight is that the interactive jobs can be served by a small pool of dedicated machines with their associated storage, while the less time-sensitive jobs can run in a batch fashion on the rest of the cluster. Those studies achieves energy efficiency by intrusive solutions, i.e., modify underlying HDFS file systems. Our work differs from these previous efforts in that we design and develop a heterogeneity aware and less intrusive approach for improving the energy efficiency in heterogeneous Hadoop clusters through adaptive task assignment.

VIII. Conclusions and Future Work

In this work we observed that the heterogeneity of both machine and workload in Hadoop clusters causes complex and time-varying energy consumption characteristics at run-time leading poor energy efficiency. We designed a heterogeneity-aware and adaptive task assignment approach, E-Ant, that improves the the energy efficiency of a heterogeneous Hadoop cluster without a priori knowledge of the workload properties. Furthermore, we addressed the challenges imposed by system noise and job fairness requirements through a rigorous system design. Our testbed implementation and extensive evaluation demonstrated the effectiveness of E-Ant. The results show that E-Ant improves the overall energy savings for a synthetic workload from Microsoft by 17% and 12% compared to Fair Scheduler and Tarazu, respectively.

In future work, we will explore the integration of E-Ant with cluster resource provisioning and DVFS/DFS techniques to further improve the energy efficiency of heterogeneous Hadoop clusters.

Acknowledgement

This research was supported in part by U.S. NSF CAREER award CNS-0844983, research grants CNS-1422119 and CNS-1217979, and NSF of China research grant No.61328203.

References

[1] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: cluster computing with working sets," in *Proc. USENIX HotCloud*, 2010.

[2] V. K. Vavilapalli, A. C. Murthy, C. Douglas, S. Agarwal, M. Konar, R. Evans, T. Graves, J. Lowe, H. Shah, S. Seth, B. Saha, C. Curino, O. O'Malley, S. Radia, B. Reed, and E. Baldeschwieler, "Apache hadoop yarn: Yet another resource negotiator," in *Proc. ACM SoCC*, 2013.

[3] Z. Abbasi, M. Pore, and S. K. S. Gupta, "Online server and workload management for joint optimization of electricity cost and carbon footprint across data centers," in *Proc. IEEE IPDPS*, 2014.

[4] J. Yao, X. Liu, W. He, and A. Rahman, "Dynamic control of electricity cost with power demand smoothing and peak shaving for distributed internet data centers," in *Proc. IEEE ICDCS*, 2012.

[5] L. Zhang, Z. Li, C. Wu, and M. Chen, "Online algorithms for uploading deferrable big data to the cloud," in *Proc. IEEE INFOCOM*, 2014.

[6] C. Reiss, A. Tumanov, G. R. Ganger, R. H. Katz, and M. A. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in *Proc. ACM SoCC*, 2012.

[7] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, and C. Hyser, "Renewable and cooling aware workload management for sustainable data centers," in *ACM SIGMETRICS*, 2012.

[8] L. E. Sueur and G. Heiser, "Slow down or sleep, that is the question," in *Proc. USENIX ATC*, 2011.

[9] Q. Zhang, F. Mohamed, S. Zhang, Q. Zhu, B. Raouf, and L. Joseph, "Dynamic energy-aware capacity provisioning for cloud computing environments," in *Proc. IEEE ICAC*, 2012.

[10] F. Ahmad, S. Chakradhar, A. Raghunathan, and T. N. Vijaykumar, "Tarazu: optimizing mapreduce on heterogeneous clusters," in *Proc. ACM ASPLOS*, 2012.

[11] "PUMA: Purdue mapreduce benchmark suite," http://web.ics.purdue.edu/~fahmad/benchmarks.htm.

[12] K. Krish, A. Anwar, and A. R. Butt, "Sched: A heterogeneity-aware hadoop workflow scheduler," in *Proc. IEEE MASCOTS*, 2014.

[13] "Watts up pro," http://www.wattsupmeters.com/,2010.

[14] R. Appuswamy, C. Gkantsidis, D. Narayanan, O. Hodson, and A. Rowstron, "Scale-up vs scale-out for hadoop: Time to rethink?" in *Proc. ACM Symposium on Cloud Computing (SoCC)*, 2013.

[15] Cloudera, "Configuration," http://blog.cloudera.com/blog/author/aaron/.

[16] H. Chen, M. K. Cheng, and Y. Kuo, "Assigning real-time tasks to heterogeneous processors by applying ant colony optimization," *Journal of Parallel and Distributed Computing*, vol. 71, 2011.

[17] Y. Wang, J. Tan, W. Yu, L. Zhang, X. Meng, and X. Li, "Preemptive reduce task scheduling for fair and fast job completion," in *Proc. USENIX Int'l Conf. on Autonomic Computing (ICAC)*, 2013.

[18] B. Cho, M. Rahman, T. Chajed, I. Gupta, C. Abad, N. Roberts, and P. Lin, "Natjam: Eviction policies for supporting priorities and deadlines in mapreduce clusters," in *Proc. ACM SoCC*, 2013.

[19] J. Wolf, D. Rajan, K. Hildrum, R. Khandekar, V. Kumar, S. Parekh, K. Wu, and A. Balmin, "Flex: A slot allocation scheduling optimizer for mapreduce workloads," in *Proc. ACM/IFIP/USENIX Middleware*, 2010.

[20] D. Cheng, J. Rao, C. Jiang, and X. Zhou, "Resource and deadline-aware job scheduling in dynamic hadoop clusters," in *Proc. IEEE IPDPS*, 2015.

[21] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica, "Improving mapreduce performance in heterogeneous environments," in *Proc. USENIX OSDI*, 2008.

[22] Q. Zhang, M. F. Zhani, R. Boutaba, and J. L. Hellerstein, "Harmony: Dynamic heterogeneity-aware resource provisioning in the cloud," in *Proc. IEEE ICDCS*, 2013.

[23] D. Cheng, J. Rao, Y. Guo, and X. Zhou, "Improving mapreduce performance in heterogeneous environments with adaptive task tuning," in *Proc. ACM/IFIP/USENIX Middleware*, 2014.

[24] Y. Chen, S. Alspaugh, D. Borthakur, and R. Katz, "Energy efficiency for large-scale mapreduce workloads with significant interactive analysis," in *Proc. ACM/USENIX EuroSys*, 2012.

[25] J. Leverich and C. Kozyrakis, "On the energy (in)efficiency of hadoop clusters," in *Proc. USENIX HotPower*, 2009.

[26] R. T. Kaushik and M. Bhandarkar, "Greenhdfs: Towards an energy-conserving, storage-efficient, hybrid hadoop compute cluster," in *Proc. USENIX HotPower*, 2010.

[27] W. Lang and J. M. Patel, "Energy management for mapreduce clusters," in *Proc. VLDB*, 2010.