LEVERAGING COST-EFFECTIVE AND GREEN-AWARE DATA PROCESSING FRAMEWORKS

[1] MUPPIDI. VAMSI PRABHAS, [2] Dr. M. SAMPATH KUMAR
[1]M.Tech Scholar, [2]Professor
Department of Computer Science and System Engineering,
Andhra University College of Engineering (A), Visakhapatnam, AP, India.

Abstract*:* Interest has been growing in powering datacentres (at least partially) with renewable or "green" sources of energy, such as solar or wind. However, it is challenging to use these sources because, unlike the "brown" (carbon-intensive) energy drawn from the electrical grid, they are not always available. This means that energy demand and supply must be matched, if we are to take full advantage of the green energy to minimize brown energy consumption. In this paper, we investigate how to manage a datacentre's computational workload to match the green energy supply. In particular, we consider data-processing frameworks, in which many back-ground computations can be delayed by a bounded amount of time. We propose Green-Aware, a MapReduce frame-work for a datacenter powered by a photovoltaic solar array and the electrical grid (as a backup). Green-Aware predicts the amount of solar energy that will be available in the near future, and schedules the MapReduce jobs to maximize the green energy consumption within the jobs' time bounds. If brown energy must be used to avoid time bound violations, Green-Aware selects times when brown energy is cheap, while also managing the cost of peak brown power consumption. Our experimental results demonstrate that Green-Hadoop can significantly increase green energy consumption and decrease electricity cost, compared to Hadoop.

Index Terms – Green energy; renewable energy; MapReduce; energy-aware scheduling; cost-aware scheduling.

I.INTRODUCTION

It is well-known that datacenters consume an enormous amount of power [31], representing a financial burden for their operating organizations, an infrastructure burden on power utilities, and an environmental burden on society. Large Internet companies (e.g., Google and Microsoft) have significantly improved the energy efficiency of their multi-megawatt datacenters. However, the majority of the energy consumed by datacenters is actually due to countless small and medium-sized ones [31], which are much less efficient. These facilities range from a few dozen servers housed in a machine room to several hundreds of servers housed in a larger enterprise installation.

These cost, infrastructure, and environmental concerns have prompted some datacenter operators to either generate their own solar/wind energy or draw power directly from a nearby solar/wind farm. Many small and medium datacen-ters (partially or completely) powered by solar and/or wind energy are being built all over the world (see http://www.eco-businesslinks.com/green web hosting.htm for a partial list). This trend will likely continue, as these technologies' capi-tal costs continue to decrease (e.g., the cost of solar energy has decreased by 7-fold in the last two decades [29]) and governments continue to provide incentives for green power generation and use (e.g., federal and state incentives in New Jersey can reduce capital costs by 60% [7]).

For the scenarios in which green datacenters are appropriate, we argue that they should connect to both the solar/wind energy source and the electrical grid, which acts as a backup
when green energy is unavailable. The major challenge with solar or wind energy is that, unlike brown energy drawn from the grid, it is not always available. For example, photovoltaic (PV) solar energy is only available during the day and the amount produced depends on the weather and the season.

To mitigate this variability, datacenters could "bank" green energy in batteries or on the grid itself (called net metering). However, these approaches have many problems:

batteries incur energy losses due to internal resistance and self-discharge; (2) battery-related costs can dominate in solar-powered systems [11]; (3) batteries use chemicals that are harmful to the environment; (4) net metering incurs losses due to the voltage transformation involved in feeding the green energy into the grid; (5) net metering is unavail-able in many parts of the world; and (6) where net metering is available, the power company may pay less than the retail electricity price for the green energy. Given these problems, the best way to take full advantage of the available green energy is to match the energy demand to the energy supply.

Thus, in this paper, we investigate how to manage the computational workload to match the green energy supply in small/medium datacenters running data-processing frame-works. In particular, we consider the MapReduce framework and its Hadoop implementation [4]. Data-processing frameworks are an interesting target for our research, as they are popular and often run many low-priority batch process-ing jobs, such as background log analysis, that do not have strict completion time requirements; they can be delayed by a bounded amount of time. However, scheduling the energy consumption of MapReduce jobs is challenging, because they do not specify the number of servers to use, their run times, or their energy needs. Moreover, a power-managing server in these frameworks requires guaranteeing that the data to be accessed by the jobs remains available.

With these observations in mind, we propose Green-Hadoop, a MapReduce framework for datacenters powered by PV solar arrays and the electrical grid (as a backup). Green-Aware seeks to maximize the green energy consump-tion of the MapReduce workload, or equivalently to minimize its brown energy consumption. Green-Aware predicts the amount of solar energy that will likely be available in the future, using historical data and weather forecasts. It also estimates the approximate energy needs of jobs using histor-ical data. Using these predictions, Green-Aware may then decide to delay some (low-priority) jobs to wait for available green energy, but always within their time bounds. If brown energy must be used to avoid bound violations, it sched-ules the jobs at times when brown energy is cheap, while also managing the cost of peak brown power consumption. Green-Aware controls energy usage by using its predic-tions and knowledge of the data required by the scheduled jobs. With this information, it defines how many and which servers to use; it transitions other servers to low-power states to the extent possible.

II. MAPREDUCE IN GREEN DATACENTERS

We propose Green-Aware, a data-processing framework for datacenters powered by PV solar panels and the electricity grid. Green-Aware relies on predictions of the availability of solar energy, and a scheduling and data availability algo-rithm that is aware of green energy, brown energy prices, and peak brown power charges. We refer to the overall brown en-ergy and power costs as the brown electricity cost.

To achieve its goals of maximizing green energy usage and minimizing brown electricity cost, Green-Aware may delay the execution of some jobs. To avoid excessive de-lays, Green-Aware attempts to complete all jobs within a bounded amount of time from their submissions. Green-Hadoop is beneficial because datacenters are often under-utilized and many jobs have loose performance requirements (e.g., data and log analysis, long simulations, jobs submitted on Friday whose output is not needed until Monday).

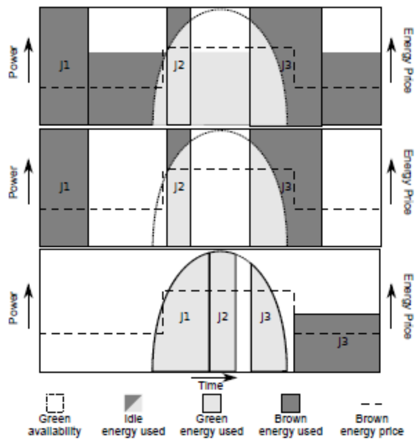

Figure 2. Scheduling 3 MapReduce jobs (J1-J3) with Hadoop (top), energy-aware Hadoop (middle), and GreenHadoop (bottom).

Figure 2 illustrates the behavior of Green-Aware (bot-tom), in comparison to conventional Hadoop (top) and an energy-aware version of Hadoop (middle) for three MapRe-duce jobs. Hadoop executes the jobs immediately when they arrive, using all servers to complete the jobs as quickly as possible. Hadoop keeps all servers active even if they are idle. As a result, Hadoop wastes substantial green and brown energy, and incurs unnecessary energy costs. In contrast, the energy-aware Hadoop that we implemented reduces waste by transitioning idle servers to a low-power state.

Green-Aware behaves differently. It uses as many servers as green energy can sustain when it is available, fewer servers (if possible) when brown energy is cheap, and even fewer (if at all necessary) when brown energy is expensive. Figure 2(bottom) shows that Green-Aware delayed jobs J1 and J2 to maximize the green energy consumption. More interestingly, Green-Aware executed part of J3 with green energy, and delayed the other part until the brown energy became cheaper. Moreover, Green-Aware did not use all servers to run J3 with brown energy to limit the peak brown power costs. When certain servers need not be fully active, Green-Aware transitions them to lower power states.

We designed Green-Aware as a wrapper around a modi-fied version of Hadoop. The wrapper implements the schedul-ing, data management, and prediction of solar energy avail-ability described in the remainder of the section. We also briefly discuss our modest changes to Hadoop.

2.1 SCHEDULING AND DATA AVAILABILITY

2.1.1 OVERVIEW

At submission time, users can specify the priority for their jobs. Like standard Hadoop, Green-Aware has five priority classes: very high, high, normal, low, and very low. Green-Hadoop executes very high and high priority jobs as soon as possible, giving them all the servers they can use. In contrast,

Green-Aware may delay some of the normal, low, and very low priority jobs by a bounded amount of time (by default, at most one day in our experiments). These behaviors reflect our performance goals: high-priority jobs should complete as quickly as in standard Hadoop, whereas low-priority ones must complete within their time bounds.

Green-Aware divides time into fixed-size "epochs" (four minutes in our experiments). At the beginning of each epoch, Green-Aware determines whether the number of active servers should be changed and whether all the data needed by the scheduled jobs is available. Specifically, servers can be in one of three states in Green-Aware: Active, Decom-missioned, or Down (ACPI's S3 state). In the Decommis-sioned state, no new tasks are started on the server, but pre-viously running tasks run to completion. Moreover, no new blocks are stored at a decommissioned server, but it still serves accesses to the blocks it currently stores. Whenever servers are not needed for computation, Green-Aware first transitions them to the Decommissioned state, and then later sends them to the Down state. To prevent data unavailabil-ity, it replicates any data needed by scheduled jobs from the decommissioned servers before sending them down. Every so often, Green-Aware reduces the amount of replication.

1) Initial configuration; 2) Green-Aware does not need one of the ac-tive servers and sends the active server with the fewest data blocks required by the jobs in the Run queue to the Decommissioned state;

Green-Aware replicates the required block from the decommis-sioned server to an active server, and then transitions the decom-missioned server to the Down state; 4) A new job is moved to the Run queue, requiring data stored only on a down server; 5) Green-Aware transitions the down server to Decommissioned to provide the needed data; 6) Green-Aware needs one more active server, transitioning the decommissioned server to the Active state; Green-Aware still needs another server, and so it activates the down server with the most data.

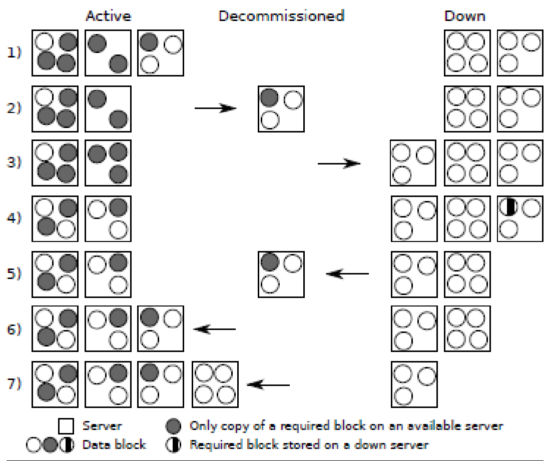
Active
Decommissioned
Down
1)
2)
3)
4)
5)
6)
7)
Server
Data block
Only copy of a required block on an available server
Required block stored on a down server

Figure 4. Server power state and data management sequence. 1)

Figure 4 illustrates an example of Green-Aware manag-ing its servers' power states and data availability. When it reduces the number of active servers (step 2), Green-Aware splits them into two groups: (a) those that have executed tasks of still-running jobs, and (b) those that have not. Then, it sorts each group in ascending order of the amount of data stored at the server that is required by jobs in the Run queue. Finally, it starts sending servers in group (b) to the Decom-missioned state, and then moves to group (a), stopping when the target reduction has been reached.

After adjusting the number of active servers, Green-Hadoop considers sending decommissioned servers to the Down state (step 3). A decommissioned server cannot go down if it has executed a task of still-running jobs. More-over, before the server can go down, Green-Aware must copy (replicate) any data that is required by a job in the Run queue but that is not currently available on the active servers.

In the opposite direction, when servers need to be transitioned from Decommissioned to Active (step 6), Green-Hadoop divides the decommissioned servers into the same two groups as above. Then, it sorts each group in decreas-ing order of the amount of data stored at the server that is required by the jobs in the Run queue. It starts activating the decommissioned servers in group (a) and then moves to group (b), stopping if the target number is reached. If more active servers are needed (i.e., the decommissioned servers are not enough to reach the target), Green-Aware activates the servers in Down state that store the most data required by jobs in the run queue (step 7).

2.2 POWER STATE AND DATA MANAGEMENT RATIONALE

Green-Aware relies on dynamic replication of data blocks to guarantee data availability when servers are turned off. However, it is possible to manage power states and data availability without dynamic replication, as in the covering subset approach [20]. This approach ensures that at least one copy of every data block stored in a Hadoop cluster is kept on a designated subset of servers (the covering subset); keeping these servers in either the Active or Decommissioned state would statically guarantee the availability of all data.

Unfortunately, using the covering subset approach would limit Green-Aware’s ability to send servers to the Down state. For example, if each data block is replicated 3 times— a typical replication level—then to keep storage balanced across the cluster, roughly 1/3 of the servers would have to be in the covering subset. The servers in the covering subset cannot be sent to the Down state, even if they are not needed for running computations.

2.3 PREDICTING THE AVAILABILITY OF SOLAR ENERGY

Green-Aware can easily use any model that predicts the availability of green energy. In fact, it would even adapt to wind energy predictions without modification. Our cur-rent implementation uses the model introduced by Sharma et al. [26] to predict solar energy. This model is based on the simple premise that energy generation is inversely re-lated to the amount of cloud coverage, and is expressed as: Ep(t) = B(t)(1 CloudCover), where Ep(t) is the amount of solar energy predicted for time t, B(t) is the amount of solar energy expected under ideal sunny conditions, and CloudCover is the forecasted percentage cloud cover (given as a fraction between 0 and 1).1

We implement solar energy prediction using the above model at the granularity of an hour. We use weather forecasts from Intellicast.com, which provides hourly predictions that include cloud coverage for up to 48 hours into the future. We use historical data to instantiate B(t). We compute a distinct B(t) for each month of the year to account for seasonal ef-fects. For each month, we set B(t) to the actual energy gen-erated by the day with the highest energy generation from the same month of the previous year. (For new installations, it is also possible to use data from the previous month.)

Unfortunately, weather forecasts can be wrong. For ex-ample, we have observed that predictions of thunderstorms are frequently inaccurate and can remain inaccurate through-out a day. Furthermore, weather is not the only factor that affects energy generation. For example, after a snow storm, little energy will be generated while the solar panels remain covered by snow even if the weather is sunny.

To increase accuracy in these hard-to-predict scenarios, we use an alternate method of instantiating CloudCover pro-posed by Goiri et al. [9]. Specifically, we assume that the recent past can predict the near future [26], and compute CloudCover using the observed energy generated in the pre-vious hour. When invoked, our prediction module compares the accuracy of the two prediction methods for the last hour and chooses the more accurate method to instantiate Cloud-Cover for the remainder of the current day. Beyond the cur-rent day, we instantiate CloudCover using weather forecasts.

2.4 MODIFIED HADOOP

As already mentioned, most of Green-Aware is imple-mented in a wrapper external to Hadoop. However, we also extended Hadoop itself with power management functional-ity. Our main changes include (1) the introduction of the De-commissioned and Down states for servers, which involved changes to the JobTracker and NameNode processes; (2) en-abling the NameNode process to know what data is present in down servers, so that they can be activated when their data is not replicated elsewhere; and (3) improvements to the block replication and replica removal functionality al-ready present in Hadoop.

III. EVALUATION RESULTS

This section presents our experimental results. First, we evaluate the accuracy of our solar energy predictions. Sec-ond, we compare GreenOnly, GreenVarPrices, and Green-VarPricesPeak in turn with Hadoop and EAHadoop to iso-late the benefits of being aware of green energy, being aware of brown energy prices, and being aware of peak brown power charges. Third, we consider the impact of various parameters, including the amount of green energy avail-able, datacenter utilization, fraction of high priority jobs, and shorter time bounds. Fourth, we study the accuracy of GreenVarPricesPeak's energy estimates. Finally, we com-pare the GreenVarPricesPeak results for FaceD and NutchI. Throughout these experiments, none of the systems we study violates any job time bounds except when we explore time bounds of 12 hours or less.

	Prediction Error (%)					
	1	3	6	12	24	48
Average	12.6	17.6	21.6	21.4	21.2	20.6
Median	11.0	16.4	20.4	20.4	20.3	17.5
90th percentile	22.2	28.9	34.0	37.4	37.9	37.8

Table 2. Error when predicting 1, 3, 6, 12, 24, and 48 hours ahead.

Predicting solar energy. Table 2 shows the percentage pre-diction error for daily energy production when predicting 1 to 48 hours ahead for the two months that include the 10 days used in our evaluation. We compute this error as the sum of the absolute difference between the predicted value and actual energy production for each hour in a day, divided by the ideal daily production (i.e., P23t=0 B(t)).

These results show that the predictions are reasonably ac-curate, achieving median and 90th percentile errors of 11.0% and 22.2%, respectively, when predicting energy production for the next hour. The predictions tend to be less accurate for Low energy days because predicted cloud cover levels are more inaccurate when the weather is partly cloudy. The pre-dictions become more accurate when the weather is mostly cloudy, as in the Very Low-Very Low days. Interestingly, while prediction accuracy drops as the prediction horizon stretches from 1 to 6 hours, beyond 6 hours accuracy some-times improves. The reason is that the accuracy of the cloud cover information tends to vary widely with time. Of the 5 pairs of days we consider, predictions are most accurate for the High-High pair, with an average 1-hour ahead prediction error of 6.1%, and worst for the Low-Low pair, with an av-erage 1-hour ahead prediction error of 23.8%.

To understand the impact of these mispredictions, we compare GreenVarPricesPeak when using our prediction vs. when using (idealized) perfect future knowledge of en-ergy production. This comparison shows that the difference in green energy consumption, and total brown electricity cost per job between the two versions is always under 16.4%. The maximum difference occurs on the "Low-Low" days. This suggests that Green-Aware could benefit somewhat from greater green energy prediction accuracy.

Scheduling for solar energy. Figures 5 and 6 show the behavior of Hadoop and EAHadoop, respectively, for the FaceD workload and the High-High days. The X-axis repre-sents time, whereas the Y-axis represents cluster-wide power consumption (left) and brown energy prices (right). The fig-ures depict the green and brown energy consumptions us-ing areas colored light gray and dark gray, respectively. The two line curves represent the green energy available (labeled "Green actual") and the brown energy price ("Brown price").

These figures show that EAHadoop successfully reduces the overall energy consumption of the workload. This effect is most obvious around noon on Tuesday. However, both Hadoop versions waste a large amount of green energy (31% for both), which could be used instead of brown energy.

In contrast, Figure 7 depicts the behavior of GreenOnly under the same conditions. In this figure, we plot the amount of green energy that Green-Aware predicted to be available an hour earlier (labeled "Green predicted"). The green pre-diction line does not exactly demarcate the light gray area, because our predictions are sometimes inaccurate.

A comparison between the three figures clearly illustrates how GreenOnly is capable of using substantially more green energy and less brown energy than Hadoop and EAHadoop, while meeting all job time bounds. GreenOnly spreads out job execution, always seeking to reduce the consumption of brown energy within resource and time constraints. Overall, GreenOnly consumes 30% more green energy than Hadoop and EAHadoop, respectively, in this experiment. Although GreenOnly does not consider brown energy prices, its brown electricity cost savings reach 30% and 29% compared to Hadoop and EAHadoop, respectively. (Note that these cost calculations do not consider peak brown power charges.)

Compared to Hadoop, the above gains come from:
(1) batching of delayed jobs, which increases server energy ef-ficiency and reduces overall energy consumption;
(2) reduc-ing idle energy by transitioning servers to low-power states; and
(3) replacing some of the remaining brown energy with green energy. Compared to EAHadoop, the gains come from sources (1) and (3), as well as the fact that batching

Scheduling for variable brown energy prices. Green-Hadoop can reduce costs further when brown energy prices vary and brown energy must be used to avoid time bound violations. To quantify these additional savings, we now study GreenVarPrices in the absence of peak brown power charges. Figure 8 shows the behavior of GreenVarPrices for FaceD and the High-High days. Comparing this figure against Fig-ure 7, one can see that GreenVarPrices moves many jobs that must consume brown energy to periods with cheap brown energy. For example, GreenOnly runs many jobs on Tuesday night that consume expensive brown energy. Those jobs get scheduled during periods of cheap energy (starting at 11pm on Tuesday and lasting beyond the 2-day window depicted in the figure) under GreenVarPrices. As a result, GreenVar-Prices exhibits higher brown electricity cost savings of 41% compared to Hadoop for the same days. So far, we have assumed that all jobs in the workload are submitted with nor-mal priority. Here, we investigate the impact of having percentages of high-priority jobs. We selected jobs to be assigned high priority randomly, according to the desired percentage. Recall that Green-Aware does not delay high-or very-high priority jobs.

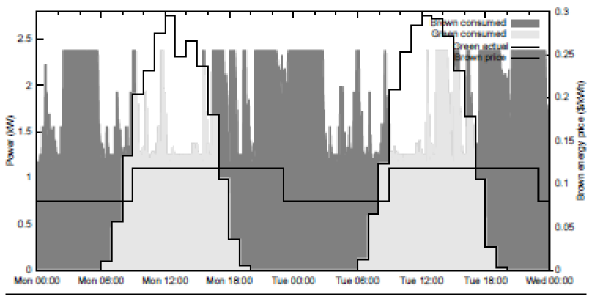

Figure 5. Hadoop for FaceD workload and High-High days.

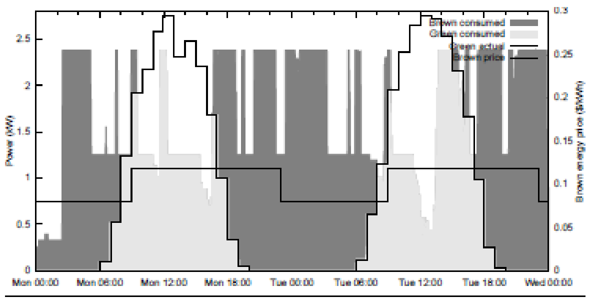

Figure 6. EAHadoop for FaceD workload and High-High days.

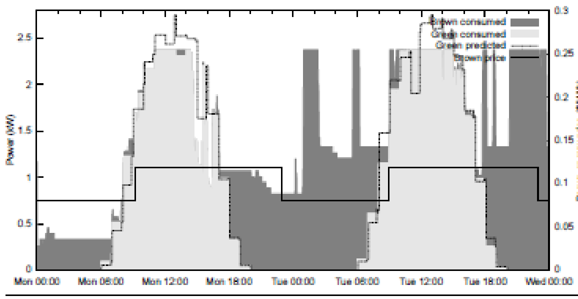

Figure 7. GreenOnly for FaceD workload and High-High days.

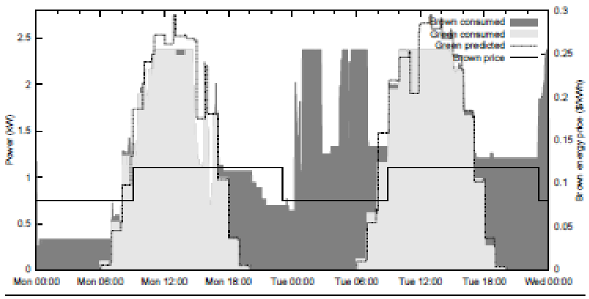

Figure 8. GreenVarPrices for FaceD and High-High days.

The experiments so far use time bounds of one day. In those experiments, Green-Aware never causes a time bound violation, nor does it need to take any action to prevent a time bound violation. We now explore Green-Aware's ability to meet tighter time bounds of 6 and 12 hours. Our results show that Green-Aware delays 19% of the jobs past their 6 hour time bounds (only 3% of the jobs are delayed by more than 20%), and 3% past their 12-hour time bounds. Note that some of the jobs in the FaceD workload take longer than 5 hours, making the 6-hour time bound tight. Further, most of the violations occur on the second day, after Green-Aware has delayed many jobs to use green energy. At that point, an additional heavy load arrives, including two very large jobs, which prevents the system from meeting all bounds.

Impact of data availability approach. We compare Green-Hadoop's data management against the covering subset approach [20]. Specifically, we run FaceD at different datacen-ter utilizations (from 13% to 92%) for the High-High days. The results show that, for low utilizations, Green-Aware re-duces brown electricity cost by more than 23% compared to the covering subset approach. Green-Aware achieves this reduction by deactivating all but one server (when possible), whereas the covering subset approach requires at least four servers to be active at all times (see Section 3.1.2). Under high utilizations, the two approaches perform similarly be-cause there are fewer opportunities to deactivate servers.

One potential drawback of our data management approach is that data may not be immediately available for arriving high-priority jobs. We run FaceD with 66% of high-priority jobs to assess this delay. The results show that all jobs start less than 6 seconds after submission; this delay can include the time to wake up to 15 servers.

Impact of workload (NutchI). Thus far, we have focused exclusively on FaceD. We have also studied NutchI to inves-tigate whether Green-Aware's benefits extrapolate to a substantially different workload. The results from these experiments support our most important observations with FaceD:

EAHadoop provides little benefit on its own; (2) Green-VarPricesPeak is able to conserve a significant amount of brown energy by batching load, transitioning servers to low-power states, and leveraging green energy; and (3) it in-creases the green energy consumption and reduces brown electricity cost significantly. For example, Green-Aware in-creases green energy use by 23% and reduces brown electricity cost by 64% compared to Hadoop for the High-High days. These results show that Green-Aware is robust to different workload characteristics.

Our results above already provide strong evidence that Green-Aware is robust to prediction inaccuracies. To further confirm this robustness, we evaluate how Green-Aware re-acts to a large, sudden change in the workload. Specifically, using the High-High days, we run FaceD during the first day and then abruptly switch to running NutchI on the second day. Right after this switch, Green-Aware's energy usage prediction per job becomes off by approximately 50%. How-ever, Green-Aware is able to adjust its energy estimation to less than 15% error within two hours, and, at the end of the two days, uses 21% more green energy than EAHadoop, re-ducing the cost of brown energy by 57%.

IV. RELATED WORK

Exploiting green energy in datacenters. Green-Aware lowers brown energy/power consumption, monetary costs, and environmental impact. Previous works have addressed some of these issues [17, 19, 22, 30]. Stewart and Shen dis-cuss how to maximize green energy use in datacenters [30]. However, their main focus was on request distribution in multi-datacenter interactive services. Similarly, [17, 19, 22] focused on these services. Akoush et al. [1] considered workload distribution in virtualized systems. Our work dif-fers in many ways. Specifically, only [22] considered green energy predictions, and only [17, 19, 22] considered vari-able brown electricity prices. None of these papers consid-ered peak power or MapReduce job scheduling. MapReduce jobs typically run longer than interactive service requests and often have loose completion time requirements, thereby increasing the opportunity to exploit green energy.

Goiri et al. [9] and Krioukov et al. [14, 15] have pro-posed green energy-aware batch job schedulers for a single datacenter. Unlike Green-Aware, however, these works re-quire extensive user-provided information (numbers of re-quired servers, run times, and completion time deadlines) for each job. Moreover, these schedulers do not manage data availability, assuming that the entire dataset of all jobs is ei-ther network-attached or replicated at every server. Green-Hadoop is substantially more challenging, since it does not leverage any user-provided information about job behavior, and explicitly manages data availability.

Aksanli et al. [2] used green energy to process Hadoop jobs that share the same datacenter with an interactive ser-vice. However, they did not consider high-priority jobs, data management, brown energy prices, or peak power charges.

Willow [12] assumes that decreases in green energy sup-ply affect the servers differently, and migrates load away
from energy deficient servers within a datacenter. In con-trast, Blink [27] considered managing server power states when the amount of green energy varies but the datacenter is not connected to the electrical grid. We argue that it is not realistic for datacenters to depend completely on green en-ergy, since this may cause unbounded performance degrada-tion. Our approach for managing green energy consumption is through scheduling, rather than load migration or server power state management.

At a much lower level, SolarCore [21] is a multi-core power management scheme designed to exploit PV solar energy. SolarCore focuses on a single server, so it is closer to the works that leverage green energy in embedded systems.

Managing brown energy prices and peak brown power charges. Most works that have considered variable energy prices have targeted request distribution across multiple dat-acenters in interactive Internet services [17, 19, 22, 24]. The exception is [18], which considers variable energy prices and peak power charges in multi-datacenter high-performance computing clouds. Our work differs in that it seeks to maxi-mize green energy use, predict green energy availability, and schedule MapReduce jobs within a single datacenter. Also within a single datacenter, Goiri et al. [9] sched-uled scientific batch jobs taking into account brown energy prices, but not peak brown power charges. Govindan et al. studied how energy stored in the UPS batteries of con-ventional datacenters can be used to manage peak power and its costs. Green-Aware targets a different type of clustered system and relies solely on software to manage peak costs.

Traditional job schedulers. Traditional batch job sched-ulers for clusters, e.g. [8, 32], seek to minimize waiting times, makespan, and/or bounded slowdown; unlike Green-Hadoop, they never consider green energy, brown energy prices, or peak brown power charges. In addition, similarly to real-time scheduling [23], Green-Aware recognizes that many jobs have loose performance requirements (i.e., can be delayed within a bound) and exploits this in favor of higher green energy consumption and lower brown electricity cost. MapReduce and Hadoop. Several efforts have sought to reduce the energy consumption of Hadoop clusters, e.g. [13, 20]. The main issue that these efforts address is how to place data replicas in HDFS, so that servers can be turned off without affecting data availability. Amur et al. addresses a similar issue in a power-proportional distributed file system, called Rabbit [3], based on HDFS. These data placement efforts could be combined with Green-Aware to reduce the need for it to replicate data to turn servers off.

V. CONCLUSIONS

In this paper, we proposed Green-Aware, a MapReduce framework for a datacenter powered by solar energy and the electrical grid. Green-Aware seeks to maximize the green energy consumption within the jobs' time bounds. If brown energy must be used, Green-Aware selects times when brown energy is cheap, while also managing the cost of peak brown power consumption. Our results demonstrate that Green-Aware can increase green energy consumption by up to 31% and decrease electricity cost by up to 39%, compared to Hadoop. Based on these positive results, we conclude that green datacenters and software that is aware of the key characteristics of both green and brown electricity's can have an important role in building a more sustainable and cost-effective IT ecosystem. To demonstrate this in practice, we are building a prototype micro-datacenter powered by a solar array and the electrical grid (http://parasol.cs.rutgers.edu). The micro-datacenter will use free cooling almost year-round and will be placed on the roof of our building.

REFERENCES

1. S. Akoush et al. Free Lunch: Exploiting Renewable Energy for Computing. In HotOS, 2011.
2. B. Aksanli et al. Utilizing Green Energy Prediction to Sched-ule Mixed Batch and Service Jobs in Data Centers. In Hot-Power, 2011.
3. H. Amur et al. Robust and Flexible Power-Proportional Stor-age. In SOCC, June 2010.
4. Apache. Apache Hadoop. http://hadoop.apache.org/, .
5. Apache. Apache Nutch, . http://nutch.apache.org/.
6. J. Dean and S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. In OSDI, December 2004.
7. DSIRE. Database of State Incentives for Renewables and Efficiency. http://www.dsireusa.org/.
8. D. Feitelson et al. Parallel Job Scheduling – A Status Report. In JSSPP, June 2004.
   Goiri et al. GreenSlot: Scheduling Energy Consumption in Green Datacenters. In Supercomputing, November 2011.
9. S. Govindan et al. Benefits and Limitations of Tapping into Stored Energy for Datacenters. In ISCA, June 2011.
   Jossen et al. Operation conditions of batteries in PV applications. Solar Energy, 76(6), 2004.
10. K. Kant et al. Willow: A Control System for Energy and Thermal Adaptive Computing. In IPDPS, May 2011.
11. R. T. Kaushik et al. Evaluation and Analysis of GreenHDFS: A Self-Adaptive, Energy-Conserving Variant of the Hadoop Distributed File System. In CloudCom, December 2010.
12. Krioukov et al. Integrating Renewable Energy Using Data Analytics Systems: Challenges and Opportunities. Bulletin of the IEEE Computer Society Technical Committee, March 2011.
13. Krioukov et al. Design and Evaluation of an Energy Agile Computing Cluster. Technical Report EECS-2012-13, University of California at Berkeley, January 2012.
14. W. Lang and J. Patel. Energy Management for MapReduce Clusters. In VLDB, September 2010.
15. K. Le et al. Cost- And Energy-Aware Load Distribution Across Data Centers. In HotPower, October 2009.
16. K. Le et al. Reducing Electricity Cost Through Virtual Ma-chine Placement in High Performance Computing Clouds. In Supercomputing, November 2011.