Towards energy-aware scheduling in data centers using machine learning

Josep Ll. Berral[1,2], Íñigo Goiri[1,3], Ramón Nou[1,3], Ferran Julià[1], Jordi Guitart[1,3], Ricard Gavaldà[2] and Jordi Torres[1,3]
Computer Architecture Department[1], Department of Software[2] (Universitat Politècnica de Catalunya) - Barcelona Supercomputing Center[3]
{berral,igoiri,rnou,fjulia,jguitart,torres}@ac.upc.edu, gavalda@lsi.upc.edu

ABSTRACT

As energy-related costs have become a major economical factor for IT infrastructures and data-centers, companies and the research community are being challenged to find better and more efficient power-aware resource management strategies. There is a growing interest in "Green" IT and there is still a big gap in this area to be covered.

In order to obtain an energy-efficient data center, we propose a framework that provides an *intelligent* consolidation methodology using different techniques such as turning on/off machines, power-aware consolidation algorithms, and machine learning techniques to deal with uncertain information while maximizing performance. For the machine learning approach, we use models learned from previous system behaviors in order to predict power consumption levels, CPU loads, and SLA timings, and improve scheduling decisions. Our framework is vertical, because it considers from watt consumption to workload features, and cross-disciplinary, as it uses a wide variety of techniques.

We evaluate these techniques with a framework that covers the whole control cycle of a real scenario, using a simulation with representative heterogeneous workloads, and we measure the quality of the results according to a set of metrics focused toward our goals, besides traditional policies. The results obtained indicate that our approach is close to the optimal placement and behaves better when the level of uncertainty increases.

Categories and Subject Descriptors

C.4 [**Performance of Systems**]: *Modeling techniques*; I.2.6 [**Artificial Intelligence**]: Learning—*Induction, Knowledge acquisition*

General Terms

Algorithms, Management, Measurement, Performance

Keywords

Machine learning, Power efficiency, Data center, Scheduling, Simulation

1. INTRODUCTION

Energy-related costs have become a major economical factor for IT infrastructures and data-centers because of the power's price escalation. Companies are now focusing more than ever on the need to improve energy efficiency. A new challenge has appeared besides the energy cost, the reduction of the carbon footprint, due to many EU regulations and campaigns demanding greener businesses. Commercial electricity consumption is a major contributor to the rising atmospheric CO_2 levels and data centers are one of the foremost parts of the problem. Energy costs are rising, data center equipment is stressing power and cooling infrastructures, and the main issue is not the current amount of data center emissions but the fact that these emissions are increasing faster than any other carbon emission. For this reason nowadays there is a growing interest in "Green" data centers and supercomputer centers [3].

In this area, the research community is being challenged to redesign data centers, adding energy efficiency to a list of critical operating parameters that already includes service availability, reliability, and performance. A large variety of power-saving methods has been presented in recent literature. Two of the most representative ones, namely workload consolidation and turning off spare servers, have been shown to be an effective way to save power. Server consolidation implies combining workloads from separate machines and different applications into a smaller number of systems. This approach solves some interesting challenges; less hardware is required, less electrical consumption is needed for server power and cooling and less physical space is required. Intelligently turning off of spare servers that are not being used is an obvious way to reduce both power and cooling costs while maintaining good performance levels.

Nevertheless, most previous proposals focus only on particular scenarios, cover only single strategies, or deal with synthetic data for some phases of the control cycle. For this reason, we propose a framework based on classical workload consolidation for reducing the power consumption of a data center executing a real dynamic workload, which covers the whole control cycle: from the acquisition of real power measures to the scheduling of the resources in the most power-efficient way according to these measures. Our approach applies some scheduling policies that reduce the number of unused machines according to the workload needs in each moment, and decide task placing and reallocation in order to compact jobs in the lowest number of machines without degrading their service level agreements (SLA). A new challenge that needs to be addressed here is providing a well-defined metric to evaluate the effectiveness of different adaptive solutions. For this, we define additional metrics in addition to power consumption to assess the quality of a given approach.

Furthermore, some scheduling information is sometimes not available or imprecise due to the user task specification or different unexpected events inherent to the system. Some decisions use information that can vary during the execution time or is heuristically obtained. When the system and the required application-level measures might be wrong or absent, we can use predictive methods to 'model' this missing information. In order to provide a better and more *intelligent* consolidation, we propose a machine learning-based method for obtaining models of application and machine behaviors that let us predict service levels before applying changes on the system, maintaining QoS while reducing energy consumption. This work is a proof-of-concept on applying new machine learning techniques in situations where information can be missing or unclear, so for now we focus on CPU dependent workloads and CPU usage timing constraints, expecting to extend the approach towards full resource representative environments and workloads. All in all, our proposal is a vertical one, involving from physical wattage measurements to workload feature prediction and simulation, and cross-disciplinary, as it touches upon very different research areas and techniques. We believe this is the path to follow to obtain intelligent and truly efficient power management strategies.

The remainder of the paper is organized as follows. Some related work and discussion of typical approaches is presented in Section 2. Section 3 describes the basics of our approach. Section 3.1 formally states the problem of scheduling using machine learning. Section 4 presents the evaluation environment including a detailed description of the simulation and Section 5 evaluates the presented approach. Finally, some conclusions and future work are discussed in Section 6.

2. RELATED WORK

Power management in cluster-based systems is an emerging topic in the resource management area. There are several works proposing energy management for servers that focus on applying energy optimization techniques in multiprocessor environments, such as [20] and [8]. Another proposal for load balancing for power and performance optimization in this kind of environment can be found in [26]. Economical approaches are also used for managing shared server resources in e.g. [9], where authors use a greedy resource allocation distributing a web workload among different servers assigned to each service. This technique demonstrates to reduce server energy usage by 29% or more for a typical Web workload. [11] proposes a hybrid datacenter architecture that mixes low power systems and high performance ones.

We propose adding smarter scheduling policies, using machine learning techniques, to dynamically turn off idle machines and reduce the overall consumption. Khargharia et al. [19] introduce a theoretical methodology for autonomic power and performance management in e-business data centers. They optimize the performance/watt at each level of the hierarchy while maintaining scalability. The authors opt for a mathematically-rigorous optimization approach that minimizes wasted power while meeting performance constraints. Their experimental results show near 72% savings in power as compared to static power management techniques and 69.8% additional savings with the global and local optimizations. Petrucci et al. [24] developed a mixed integer programming (MIP) formulation to dynamically configure the consolidation of multiple services/applications in a virtualized server cluster. The approach is power efficiency centered and takes into account the cost of turning on/off the servers. However, it is too focused in Web workloads and the decision algorithm is intended to run every 5 minutes, in contrast with our approach, that can handle heterogeneous workloads and adapt the system at every new job arrival, making better use of energy. The use of heavy mathematical calculus in the scheduling can lead to a too slow decision process for an online scheduler like the one we look for. Other approaches dealing with uncertainty are [34], where statistic methods based on correlation are used to predict usage and so consolidate works. They measure SLA but do not take it directly into account when consolidating tasks.

The use of virtualization for consolidation is presented in [25], which proposes a dynamic configuration approach for power optimization in virtualized server clusters and outlines an algorithm to dynamically manage the virtualized server cluster. Following the same idea, [21] aims to reduce virtualized data center power consumption by supporting VM migration and VM placement optimization while reducing the human intervention, but no evaluation is provided. Other work [33] also proposes a virtualization aware adaptive consolidation approach, measuring energy costs executing a given set of applications. They use correlation techniques in order to predict usage, while we use machine learning to predict application power and performance. Also at this moment they do not apply powering off techniques, just analyze the system.

The use of heterogeneous workloads leads to SLA's where some of the applications in the system can have stringent conditions to be met. There have been several proposals into resource capacity planning and dynamic provisioning issues for QoS control (e.g. [6, 27, 29]). [10] states that new power saving policies, such as DVFS, or turning off idle servers can increase hardware problems as well as the problem to meet SLAs in this reduced environment. Following this idea, we show how scheduling policies can take into account such problems.

Machine learning approaches have also been used to reduce power consumption in clusters. Tesauro, Kephart et al. [31, 18] present a reinforcement learning approach to simultaneous online management of both performance and power consumption. These approaches look at learning what policies should be applied given a system status. Such policies save more than 10% on server power while keeping performance close to a desired target. Das et al. [14] present an approach using multi-agents in order to turning-off servers under low-load conditions, achieving 25% power savings without incurring SLA penalties on server farms. All these approaches use reinforcement learning in order to learn management policies from given data, while we are using, at this moment, induction learning to model the data for a given policy. This lets us plan, in our future work, the use of these policy learners to add a new level of adaptability to our system.

Filani et al. [15] offer a solution that includes a platform resident Policy Manager which monitors power and thermal sensors and enforces platform power and thermal policies. They explain and propose how the PM can be used as the basis of a data center power management solution. Although our scheduler does not take into account the thermal information, the turning off of servers will reduce the cooling needs.

3. ENERGY-AWARE MANAGEMENT

Our approach uses two different mechanisms in order to reduce the power consumption of a data center while respecting the different SLAs. One of the mechanisms that allows saving more power is turning off idle machines, which saves more than 200W in testbed machines. A complementary mechanism is trying to execute all the tasks but with the minimum amount of machines, known as consolidation. Therefore, scheduling takes a main role in order to achieve this power consumption reduction.

We want to turn off some idle machines in order to save power and we turn on them again if they are needed when a peak load occurs. For this purpose, our strategy is based on consolidating a set of tasks, distributed among a set of machines, into as few machines as possible without degrading excessively the execution of these jobs. Here, several scheduling policies could be applied in order to assign new jobs in the system to available machines and redistribute jobs being executed in order to make some machines idle and then turning them off [17]. Notice that turning on machines again is not a free and instantaneous process and this overhead, which can take more than a minute, must be taken into account.

We consider several traditional scheduling policies, including; *Random* which assigns the tasks randomly (taking into account if the node fits there); *Round Robin* which assigns a task to each available node, which implies a maximization of the amount of resources to a task but also a sparse usage of the resources; *Backfilling* which tries to fill as much as possible the nodes, thus solving the former problem; *Dynamic Backfilling* which is able to move (i.e. migrate) tasks between nodes in order to provide a higher consolidation level. When tasks enter or exit the system, it checks if any tasks should be moved to other nodes according to different parameters such as the system occupation, current job performance, or expected user SLA satisfaction.

While *Dynamic Backfilling* performs well when having precise information (as shown in the evaluation), other policies are necessary when information is incomplete or imprecise. For this reason, a machine learning policy is introduced in order to predict features that will only be known in the future. This lets us anticipate the SLA degree and the power consumption before placing or moving jobs, and therefore choose a job configuration that is expected to be good.

3.1 Machine Learning approach

The closely related Machine Learning and Data Mining areas are concerned with obtaining knowledge from data. This typically involves creating models or discovering patterns in examples from the past of a system behavior, with as little expert intervention as possible. In this study, we use machine learning techniques in order to predict, from our set of machines and set of jobs, the resulting client satisfaction level of each job and power consumption before placing tasks in machines or moving tasks across machines. These predictions are then used by a move selection algorithm to choose destination machines with good resulting client satisfaction and opportunities for consolidation.

For this prediction process, we need to choose suitable predictor algorithms, computationally light but able to obtain good results once trained with data from various workloads. Also, we need to obtain a good training set (a set of data containing labeled instances from representative executions) and another test (or validation) set. If, after training, the predictors' guesses are close to the correct values on the test set, we expect that they will also be correct on future real workloads. Figure 1 shows the basic schema of a supervised machine learning process.

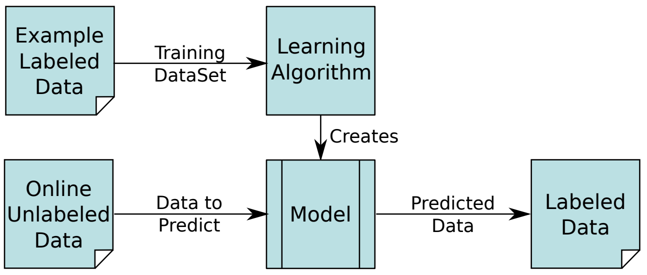

Figure 1: Supervised Machine Learning Schema

The machine learning aided policy implements a *Dynamic Backfilling* scheduler replacing the static decision maker, using the information provided directly by the user, and using as decision maker the results of the performance and power consumption estimators. This is, instead of fitting jobs in host machines directly from the user specifications, we estimate the impact the job will cause in the potential host machine, in performance parameters and power consumption.

In the line of *Dynamic Backfilling*, for each reschedule we attempt to empty low-used host machines fulfilling nearly fully-booked ones. Then, for each movement we estimate whether the job will interfere in the resource requirements of all other jobs in the machine, and the estimated new power consumption of this machine will compensate the possible performance degradation. This permits to obtain a more adaptive and robust system, where user or application specifications can be imprecise or change over time.

Currently we are assuming a negligible operation cost, but for our future work we are working out on taking factors like moving machines cost into account. Also *Dynamic Backfilling* is a costly algorithm specially when using data collection processes, so we are planning the use of some AI planning and reinforcement learning techniques [30, 32], in order to make decisions in a more accurate and not so costly way.

3.2 Relevant factors and basic assumptions

When a new job arrives, the system will try to allocate it to some host, and then perform a scheduling round in order to find a more efficient schedule. The candidate moves are of the form "move job j from its current host to host h", and the chosen one will be the one with maximum expected benefit. This benefit is the combination of two factors: the future performance of the jobs and power consumption in the resulting allocation, that we call R and C. Given a host, R_h and C_h cannot be known beforehand in general, so we will predict these values from our learned models obtaining the estimated $\hat{R_h}$ and $\hat{C_h}$.

The factor R_h indicates the *health status* of the jobs running on a machine h. This factor can be represented as a number between 0 and 100; a value close to 0 will indicate unacceptable performance, and a value close to 100 will indicate a good performance of the jobs in the machine. For this case of study, as a proof of concept we will assume that R_h depends only, for each job allocated to h, on the particular deadline constraint, indicating the SLA fulfillment.

Usually an SLA is an agreement on application resource consumption or performance guarantees (bandwidth, disk and CPU quotas, response time or throughput, time deadlines). In this paper we use a time deadline metric as SLA guarantee. The SLA fulfillment level follows a grid client satisfaction ratio, where it is fulfilled when the task completion time takes less than the deadline given by the user.

In immediate future work we are including other relevant metrics into the SLA objectives, such as throughput constraints for service applications and time of response for interactive applications.

We can define a *finished job j* by a tuple

$$j =< UserT_j, SLAFactor_j, StartT_j, EndT_j >$$

where $UserT_j$ is the user estimation of the time to complete the job, $SLAFactor_j$ is the factor over $UserT_j$ that the user is willing to accept, and $StartT_j$ and $EndT_j$ are the times in which the job was started and finished.

The performance factor R_h can be calculated in the way of

$$R_j = f(UserT_j, SLAFactor_j, StartT_j, EndT_j)$$

where function f, which is negotiated with the user, indicates the penalty for not satisfying the user's requirement, and we use it to define the fulfillment of job j, R_j (independently of the machines in which it has been executed). A very strict function f (f_{hard}) would indicate maximum loss when the SLA is not totally satisfied, while softer functions (f_{soft}) could go from 100 to 0 smoothly.

$$\begin{aligned} f_{hard} &= \begin{cases} 100 & \text{if } EndT_j - StartT_j \leq UserT_j \cdot SLAF_j \\ 0 & \text{otherwise} \end{cases} \\ f_{soft} &= max(100, \frac{UserT_j}{EndT_j - StartT_j} \cdot SLAF_j \cdot 100) \end{aligned}$$

At this stage of the work we use the softer version of f, expecting to use more elaborated functions when we dispose of more complex workloads with complex SLA requirements. For this work, the value of R_h given a machine h should be the aggregation of the values R_j for all allocated jobs on h. Supposing an initial hypothesis of fairness between the jobs on a machine, for this version of our work we take as aggregation function g the arithmetic mean of the R_j's $g(h) = \sum_j^{Jobs_h} R_j / (Num\ Jobs_h)$

The consumption factor C_h indicates the power consumption of machine h. It can be measured empirically for the training data sets, and possibly during the execution. Our experiments and common knowledge indicates that it depends mostly (but is *not* linearly proportional to) the percentage of CPU usage at h.

The global function that the system should optimize is a combination of the aggregated levels of SLA fulfillment and the total power consumption, that is of $R = g(R_1, \ldots, R_H)$ and $C = \sum_{h=1}^{H} C_h$ if we have H host machines. For the moment, we decided to choose moves that maximize R under the condition that they do not increase C; this maintains SLA accomplishment as a priority over consumption, which is the usual practice up to date.

3.3 Data sets and prediction algorithms

The values of R_h and C_h as described above are usually not known before performing the job allocation and finishing the running jobs in the machine. But we can run representative executions in order to obtain examples of $\langle jobs, machine, R_h, C_h \rangle$ configurations, and learn a model capable of predicting, from $\langle jobs, machine \rangle \Rightarrow \langle \hat{R}_h, \hat{C}_h \rangle$

To predict the power consumption $\hat{C}_h$, we found it useful to predict first the percentage of CPU usage at h. We used the Linear Regression algorithm [35], where the most relevant attributes resulted to be the CPU usage for each individual job in h and, with smaller weight, the number of jobs in the candidate host. This attribute choice was to be expected. Predicting power consumption is more complex that a simple linear regression, because it has a nonlinear relation with CPU usage, so we used the more sophisticated M5P machine learning algorithm [35]. M5P builds a decision tree splits on attributes at inner nodes and applies different linear regression at the leaves. It therefore computes a piecewise linear function of the attributes, which is enough to approximate the nonlinear dependence of power consumption on (mostly) CPU usage and number of jobs.

The real problem is predicting the deadline fulfillment of a given job, R_j because the most important value $EndT_j - StartT_j$ will not be known until the job ends, and also the user estimate $UserT_j$ may be inaccurate. Using the learned model we predict $\hat{R}_j$ using as known information: the amount $UsageCPU_j$ of CPU used by the jobs on h (included the new job), time spent so far $Now - StartT_j$ and the characteristics of the machine where it is executing $AvailableCPU_h$. For this prediction we have used another linear regression function, where the most relevant values are the timing values for j and other jobs in the same machine. In this preliminary work, assuming that all the machines have identical capacity, some attributes about machine characteristics need not to be included into the learned model, but as see during the experiments of the Linear Regression, a coefficient exists directly related to the capacity of the machine.

The algorithm pseudocode is shown in Algorithm 1. Basically it performs a dynamic backfilling strategy, using the learned model to decide whether the new allocation of the job will bring an energetic improvement given an estimated performance cost. At each scheduling round, the underused hosts are selected to be emptied, and their jobs are *virtually* allocated in nearly-full hosts. Using the model, we can estimate if this is a suitable allocation. If it is, and we consider the host can be emptied, we proceed to perform the jobs reallocation. Note that it is a greedy algorithm that is not guaranteed to find the theoretically best possible list of movements.

Algorithm 1 Algorithm for move selection

```
Poll hosts for information about their jobs and status;
OH := select "Emptiable Machines" [jobs < 4];
For each Machine (oh) in OH do:
    For each Job (j) in oh do:
        CH := select "Fillable Machines" [enough CPU and mem];
        For each Machine (ch) in CH do:
            -- predict effect of moving j from oh to ch;
            predict R(oh) and R(ch) after movement;
            predict C(oh) and C(ch) after movement;
            compute global R and C after movement;
        End For
        Get ch leading to highest R among those that decrease C;
        add movement (j,oh,ch) to List_of_movements;
    End For
    If (all jobs in oh can be reallocated) then:
        proceed with the List_of_movements;
    End If
End For
```

4. SIMULATION AND METRICS

Simulation consists of the evaluation of a set of nodes which are stressed by a given workload. The system performance is evaluated according to a set of different metrics that take into account consolidation and power consumption.

The simulation is able to compare different scheduling policies. They are implemented in a modular way and can be plugged on top of other architectures, such as EMOTIVE [16] or XtreemOS [2, 12] (inside its component Application Execution Management, AEM [13]), where a set of schedulers can be introduced and selected by the administrator. XtreemOS follows a scheme of local scheduling rather than a global one, but its direct interaction with the kernel and its dynamic resource discovery via DHT can provide a way to simplify the switch on-off techniques.

4.1 Simulation and power models

In this section we present a framework for evaluating the power efficiency of a data server which executes an heterogeneous workload. This framework tackles the whole problem from the power consumption measurement of a single machine to the execution of different applications on a data center which allows evaluating different approaches such as dynamic turn on/off or consolidation. It is based on a simulator in order to evaluate the performance of a whole data center focusing on power consumption. It allows obtaining different metrics of the modeled cluster while applying different workloads in order to optimize different policies.

Figure 2 shows the development cycle of the simulator used in our framework. Firstly, different applications with different typologies and profiles are executed (on a real, not simulated, machine) and their resource usage and power consumption monitored. Power usage is recorded using an external device which monitors the whole machine energy consumption. From these recordings, a model of the machine is built, which is then used to simulate a data center with many (identical) machines. Validations are applied to refine the model and the simulator. Finally, the simulator is executed to provide the experimental data described here.

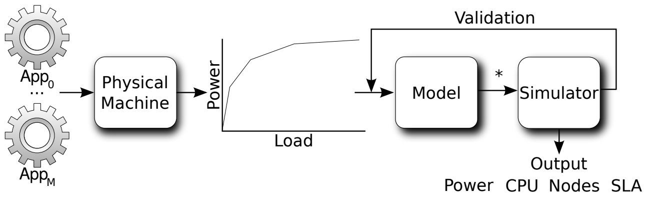

Figure 2: Simulator workflow

This simulator is able to add and remove nodes dynamically including the boot time and load times. It allows using different scheduling techniques which can take advantage of different capabilities such as migration of running tasks between nodes.

Finally, the simulator is able to read a workload and apply the different scheduling policies to output the results of executing the workload, including power consumption and other resource usage statistics, as shown in Figure 3.

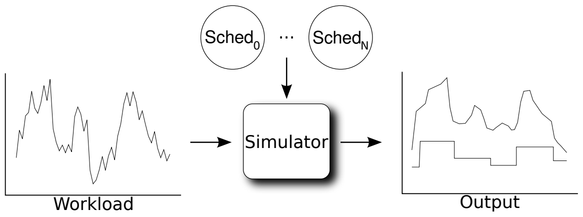

Figure 3: Simulator

4.1.1 *Energy consumption measurement*

We measured the real power consumption using different workloads in a 4-CPU computer whose kernel includes some energy efficiency policies. The power consumption of the machine was gathered using digital methods via an ammeter. In the past, analogical methods via oscilloscope where used, as seen in [22], but similar results are obtained with the ammeter method (however, instantaneous wattage is lost; we can only measure stable workloads). The resolution of the measurements is below 1 Watt. Figure 4 shows the system behavior; we can see that wattage increases with the workload (in a non constant slope), but that it is noticeable even in an idle machine, which is the main reason why we can gain by consolidation. This graph was included in the simulator, as part of the model. It is important that idle wattage level should be decreased in the industry as it is one of the most used states and it is not energy efficient, as seen in [7].

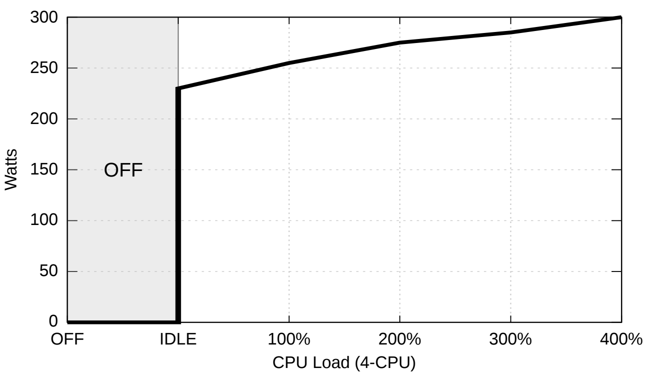

Figure 4: Power behavior of the target PC

4.1.2 Testbed simulator

Our simulation technology is based on the OMNeT++ [4] platform. The simulation uses similar techniques as seen in [23], but centered on power usage (measured in a real machine as seen in the environment section) and the scheduling of the different CPUs in the machines.

The simulation takes a list of tasks and tries to execute them in a PC containing a set of CPUs, with a specified quantum. The machines can be shutdown and booted on-demand too. To make the booting more realistic we specify a booting time of 60 seconds and a power usage as measured in the real machine.

As in [23], the simulator does not try to simulate exact execution times working step-by-step, as this would be very time consuming. It is much simpler but enough for our purposes if we can predict the system's general behavior, in the long run, for the different scheduling techniques in large clusters, with respect to Service Level Agreement and power usage. See [23] for details.

4.1.3 Model validation

We have validated the model using special schedulers whose outcome we can predict separately and testing that the simulation obtains the expected relative values (such as SLA = 1.0 or similar). Moreover we tested that simulating one machine produces similar power usage values as in the real case. Future work may include fine-tuning and a more detailed validation, as some smaller overheads and other issues that can show in the real world should be modeled and introduced in the simulator.

4.2 Metrics

One of the key proposals of this paper is the ability to compare different techniques for efficient power usage. Currently, there is no standard approach for measuring the power efficiency or the consolidation of a data center. For example, [28] proposes a benchmark for measuring power efficiency on a set of different scenarios such as mobile devices or servers. However, it does not address consolidation.

In this paper, we introduce some metrics to compare adaptive solutions. To this end, the energy consumption must be evaluated precisely. This part is mandatory to be able to compare different approaches. Nevertheless, it is not enough since a given policy can decrease the efficiency energy but it can make some tasks violate their SLAs. In addition, consolidation factors are also important for measuring the scheduling policy quality as understanding what a scheduler is doing is not easy just evaluating energy or SLA fulfillment. For this purpose, we add some other metrics that would help to comprehend and measure different relevant aspects.

Working nodes The number of nodes that are executing some task. Hence, in order to allow shutting down more nodes, less working nodes are better.

Running nodes The number of nodes that are turned on. Having a lower number of these machines is one of the key issues for saving energy in order to reduce the idle machine consumption.

CPU usage The amount of CPU time that has been used.

Power consumption Total energy consumed by the nodes.

SLA fulfillment level The client satisfaction based on the task SLAs. We evaluate a service by its availability ratio, which is 100 if it is always available, and 0 if it never is. On the other hand, we will use the typical grid client satisfaction ratio, which is 100 if execution time is less than expected time and 0 if completing the task takes longer than twice the expected time. This is defined by this equation:

$$S = \begin{cases} 100 & \text{if } ExecT < ExpectedT \\ 100 \cdot max\{1 - \frac{ExecT - ExpectedT}{ExpectedT}, 0\} & \text{if } ExecT \geq ExpectedT \end{cases}$$

5. EVALUATION

In this section, we present the experimentation regarding our strategy and the different used techniques, and also the simulation and power consumption models used for evaluating them.

5.1 Experimental environment

The experiments will consist of the simulation of a whole data center with 400 nodes that will execute different workloads and will evaluate its behavior according to different metrics including power consumption.

The presented approached intends to take benefit of the variation and the heterogeneity in current data centers. For this reason, the evaluation includes two different workloads: Grid and service oriented. The former is a Grid workload obtained from Grid5000 [5] on the week that starts on Monday first of October of 2007. The training of the ML model has been performed using the workload corresponding to the week of third of September. For evaluating the *SLA satisfaction*, SLAs have been added to the Grid jobs, specifying tolerance factors in execution times in the range 1.1. . . 2.0.

The latter workload results from the aggregation of different services based on the load of *Ask.com* [1]. These services correspond to three different profiles. One that represents a single day execution from 0:00 to 23:59 with a low usage during the night and a classical increase at the start of day. The second one follows the same behavior but it has a bigger load in the afternoon. The third uses a whole week in order to represent the weekend user decrease.

Finally, the evaluation also includes a mix of the already presented workloads in order to simulate a heterogeneous data center and test the functionality of the approaches with a realistic approach for current data centers.

5.2 Power vs. SLA fulfillment trade-off

In our approach, one of the key decisions is determining when a node should be turned off in order to save power consumption or when to turn on it again in order to be used to fulfill the tasks SLAs. This decision is driven by means of two thresholds: the minimum *Working nodes* threshold λ_{min}, which determines when the provider can start to turn off nodes, and the maximum *Working nodes* threshold λ_{max}, which determines when the provider must turn on new nodes. Finally, in order to set a minimum working set, the minimum amount of machines $\min_{exec}$ is also specified.

The effect of these two thresholds has been tested by executing the Grid workload on top of the simulated data center following the *Dynamic Backfilling* policy, which is the one which makes a more aggressive consolidation without taking into account the task SLA. This allows evaluating the influence of the turning on/off thresholds by showing the SLA and the power consumption respectively.

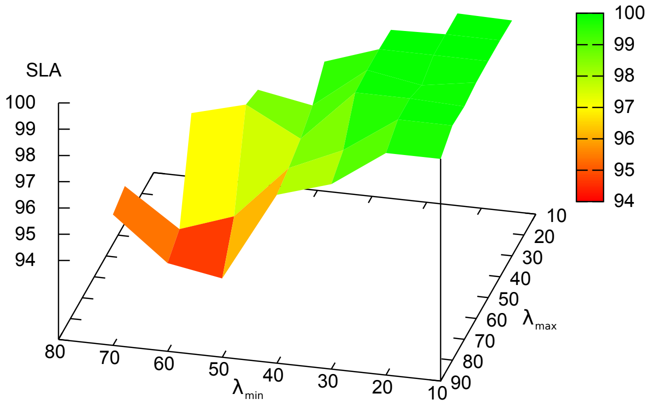

Figure 5: SLA satisfaction using different turn on/off thresholds

Figure 6 shows that waiting the nodes to reach a high utilization before adding new nodes (high λ_{max}) makes the power consumption smaller. In the same manner, the earlier the system shutdowns a machine (high λ_{min}), the smaller the power consumption is. It demonstrates how turning on and off machines in a dynamic way can be used to dramatically increase the energy efficiency of a consolidated data center.

On the other hand, SLA fulfillment decreases, as shown in Figure 5, when the turn on/off mechanism is more aggressive and it shuts down more machines (in order to increase energy efficiency). Therefore, this is a trade-off between the fulfillment of the SLAs and the reduction of the power consumption, whose resolution will eventually depend on the service provider interests.

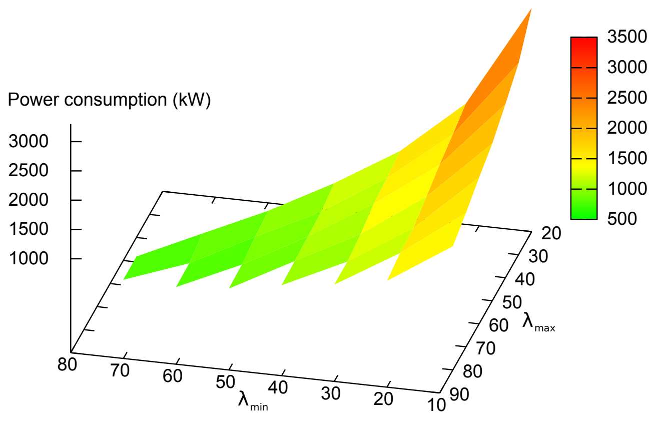

Figure 6: Power consumption using different turn on/off thresholds

Fortunately, average threshold values give a balanced trade-off between energy and SLA. According to this, in the evaluation we will use $\lambda_{min} = 30\%$ and $\lambda_{max} = 60\%$ in order to ensure almost complete fulfillment of the SLAs while getting substantial power consumption. A next step would be to dynamically adjust these thresholds, and is part of our future work.

5.3 Validation of ML models

In this section, we evaluate the accuracy of the machine learning models derived during the training process, to assess their reliability to predict future situations. Furthermore, we evaluate the performance of our overall method for scheduling and consolidating.

The role of the machine learning methods is to provide predictions of values that are unknown at the time in which they are needed. In our setting, they provide some of the inputs to the *Backfilling* and *Dynamic Backfilling* algorithms that they need to perform their scheduling, namely, anticipated power variation and SLA fulfillment resulting from a possible move. Luckily, at the validation stage, the information about actual power variation and SLA fulfillments can be read from the available datasets. We can thus evaluate predictor accuracy on a test/validation subset, disjoint from the training set.

The linear regression model to predict SLA fulfillment ratio fits with the real measurements with average accuracy close to 0.985. This very high value is explained in part by our current choice of priorities. Since we prioritize SLA fulfillment over consumption, the algorithm's choices are conservative or cautious with respect to SLA's, which are therefore almost always fully satisfied, and therefore easy to predict. In fact, we did not find situations where the predicted SLA value is 1 but the actual SLA is lower: the 0.015 fraction of prediction errors are on the side of SLA's that are predicted to fail but finally succeed. This will generally implied that our algorithms do well on the SLA side at the expense of somewhat higher than necessary power consumption.

The model for predicting CPU usage, basically using the CPU usage of all jobs allocated to it, is accurate up to 0.997, almost perfectly. CPU usage prediction is in turn used to predict the power consumption of a machine after adding a job, and we obtained a high accuracy of 0.98 between the model and the workload data, so the model is able to predict consumption for low loads, average loads, and high loads.

Having so validated the models, we can use them to provide inputs to the ML-based scheduler. Next subsection shows the results of this and other schedulers considered.

5.4 Scheduling policies

This experiment evaluates the behavior and performance of the different scheduling policies using three different workloads, namely a Grid workload, a Service workload, and a Heterogeneous workload. It uses the turn on/off thresholds $\lambda_{min} = 30\%$ and $\lambda_{max} = 60\%$ derived in Section 5.2.

We have evaluated five scheduling algorithms: *Random* and *Round-Robin* do not use any user-provided information about the jobs and do not consolidate. For *Backfilling* and *DynamicBackfilling*, the user provides for each job a figure indicating which % of a CPU capacity should suffice to satisfy the task SLA's. The algorithms trust this figure as totally reliable, and therefore will make decisions that may fit very tightly the SLA's and therefore save power. Our algorithm, *Machine Learning*, has the drawback with respect to these algorithms that it does not use any user-provided information. Therefore, a priori we should expect it to perform worse in general, as it has to pay a price for this lack of information, but the closer in performance it is to these two algorithms with privileged information, the more successful we can consider our approach. Somewhat surprisingly, we will see that it does sometimes better than the algorithms having additional information. The results are presented in Table 1, according to the metrics proposed in Section 4.2.

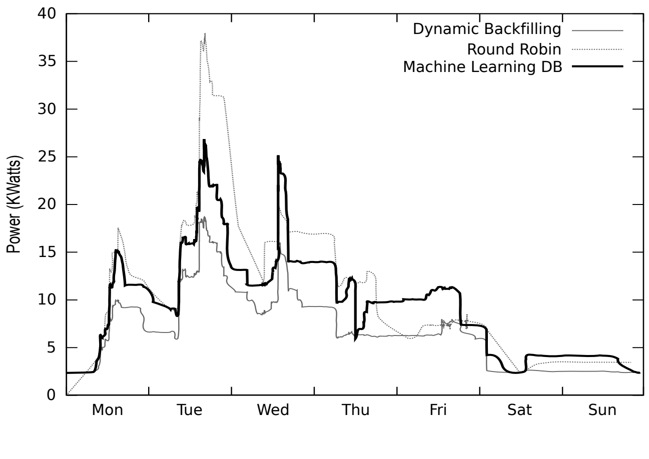

Figure 7: Power consumption of different schedulers with a Grid workload

The results obtained using the Grid workload show that non-consolidating policies such as *Random* and *Round-Robin* give a poor energy efficiency while violating some SLAs: these policies give the worst results on both criteria. *Backfilling* and *Dynamic Backfilling* fulfill all SLA's with substantially lower cost. *Machine Learning* performs almost perfectly w.r.t. SLA's (as we have seen that predictions for SLA fulfillment are very accurate), but with respect to power is closer to *Random* than to the backfilling algorithms. The reason is that the user-provided figures for the tasks are very close to the real ones (and the load quite steady), so the backfilling algorithms will take many decisions that will not violate any SLA but that look too risky to *Machine Learning*, that pays a high price in consumption for its caution.

Notice that this workload makes a very variable use of the power consumption over time as it is graphically shown in Figure 7. This is due to the fact it makes the system creating and destroying many VMs, which implies a high variability in the number of running nodes and power consumption during time. The figure shows the power consumption pattern of the different schedulers and enforces the table results.

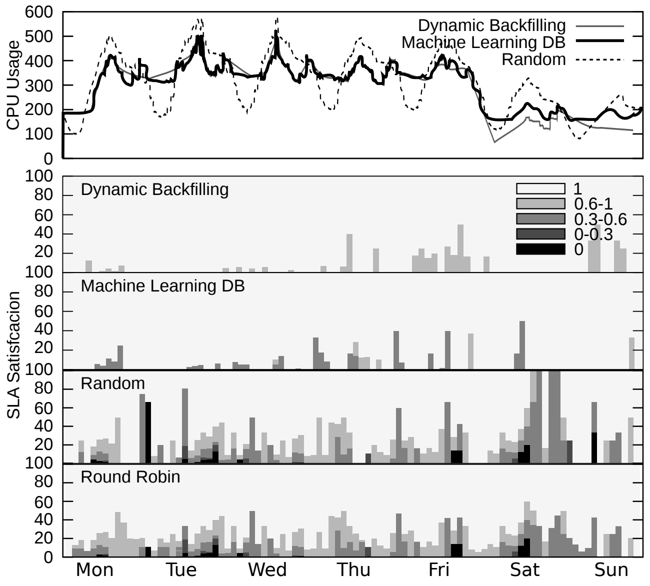

Figure 8: CPU usage and SLA fulfillment with heterogeneous workload; Most significant policies: Dynamic Backfilling, Machine Learning and Random

On the Service workload, the *Machine Learning* scheduler is the clear winner with respect to energy consumption. Note first that on this workload all the schedulers executed all the tasks, so all SLA's are fulfilled. The workload has a very variable CPU usage. This means that the user-provided estimation about the CPU to be used for the given jobs will be a large overestimation for large periods (while it was very tight on the Grid workload), and power will be unnecessarily wasted. Here is where the *Machine Learning* scheduler takes advantage because of the capability of computing somewhat conservative but adaptive estimates of the degree of SLA fulfillment, and adapt its power consumption accordingly. Thus, it is able to work better when the features of the input load are not known or the user-provided estimates are misleading, which is very often the case.

Finally, the results obtained using the Heterogeneous work-

	Working nodes (avg)	Running nodes (avg)	CPU usage (hours)	Power (kW)	SLA (%)
Grid workload					
Round Robin	16.11	41.37	5954.91	1696.66	85.99
Random	16.51	40.76	6017.85	1671.16	88.38
Backfilling	10.18	27.10	6022.34	1141.65	100.00
Dynamic Backfilling	9.91	26.46	6104.33	1118.86	100.00
Machine Learning DB	15.04	37.92	6022.27	1574.78	99.69
Service workload					
Round Robin	290.99	400.00	78419.97	19761.54	100.00
Random	218.46	400.00	75336.88	19784.38	100.00
Backfilling	108.79	352.88	59792.09	16257.26	100.00
Dynamic Backfilling	108.79	352.88	59748.10	16229.22	100.00
Machine Learning DB	99.61	270.50	61379.38	13673.71	100.00
Heterogeneous workload					
Round Robin	260.66	400.00	84432.96	19713.72	94.20
Random	224.08	400.00	82137.27	19763.63	88.53
Backfilling	110.85	330.19	65894.46	16304.38	99.50
Dynamic Backfilling	111.03	329.07	66020.58	16214.49	99.59
Machine Learning DB	124.20	307.89	68554.01	15110.33	98.63

Table 1: Scheduling results.

load are, as expected, a mix of the two previous workloads. In this case, the overall SLA fulfillment by our algorithm is worse by about 1%, but its overall power consumption is better by about 10%. Figure 8 shows the evolution over time, and one can see that *Machine Learning* does worse w.r.t. SLA when the CPU utilization is higher (i.e., when the other algorithms can exploit the user-provided information they have), but much better than Random and Round Robin, which behaves very similar to Random.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have presented a framework that provides a vertical and intelligent consolidation methodology to deal with uncertain information keeping in mind performance and power consumption at the same time. This framework covers the whole control cycle of a real scenario with a holistic approach that requires a collaboration among researchers from different disciplines. The results obtained in this paper indicate that significant improvements can be achieved using machine learning models in order to predict application SLA timings and decide the movements and operations to be done within scheduling functions.

Our experiments, performed using real workloads, exemplify that these techniques can offer substantial improvements in energy and performance efficiency in these scenarios. The experiments using the Grid workload demonstrate how non-consolidation aware policies give a poor energy efficiency. Backfilling gets a good performance and its dynamic extension demonstrates power efficiency in order to reduce power consumption, but only if reliable a priori information on the tasks is available, and if the task features are steady over time. The machine learning method is close enough to these models that use external information w.r.t. SLA fulfillment (performance), and much better with respect to power consumption when the information provided by the user is not uniformly accurate. On mixed, heterogeneous workloads, it obtains noticeable reductions in power consumption at the expense of only a slight decrease in performance.

On this work, as a proof of concept, we used a greedy algorithm for scheduling (Dynamic backfilling) and we departed from basic attributes (CPU Usage, Timing SLAs), as a first approximation of a decision making methodology. We are now focusing our work toward Reinforcement Learning algorithms, in order to optimize the search space of scheduling solutions, and be able to make decisions taking into account immediate results of job scheduling and also the consequences of choosing an specific action or another on future system configurations. Therefore, as being proved the viability of adding ML techniques to improve power management, we are now including the concept of *resource* aggregation of not only CPU but also memory and IO, and expanding the concept of SLA.

The rest of our future work will focus on the extension and addition of new modules to the simulator in order to get more information and simulate detailed scenarios such as virtual environments, and migration operational costs. Furthermore, new enhancements to the scheduling policies such as dynamic thresholds when turning on/off machine will be added.

Acknowledgment

This work is supported by the Ministry of Science and Technology of Spain under contract AP2008-0264, TIN2008-06582-C03 and TIN2008-06582-C03-01, by the Generalitat de Catalunya under contracts 2009-SGR-980 and 2009-SGR-1428, and the European Union under contract TIN2007-60625, FP6 IST contract 033576 (XtreemOS) and EU PASCAL2 Network of Excellence (FP7-ICT-216886).

7. REFERENCES

[1] Ask.com. `http://www.ask.com`.
[2] XtreemOS European Project, 2006-2010. `http://www.xtreemos.eu`.

[3] Green Grid Consortium, 2009. http://www.thegreengrid.org.

[4] Omnet, 2009. http://www.omnet.org.

[5] The Grid Workloads Archive, 2009. http://gwa.ewi.tudelft.nl.

[6] K. Appleby, S. Fakhouri, L. Fong, G. Goldszmidt, M. Kalantar, S. Krishnakumar, D. Pazel, J. Pershing, and B. Rochwerger. Oceano-SLA based management of a computing utility. In *7th IFIP/IEEE International Symposium on Integrated Network Management*, volume 5. Citeseer, 2001.

[7] L. A. Barroso and U. Hölzle. The case for energy-proportional computing. *Computer*, 40(12):33–37, 2007.

[8] R. Bianchini and R. Rajaniony. Power and energy management for server systems. *Computer(Long Beach, CA)*, 37(11):68–76, 2004.

[9] J. Chase, D. Anderson, P. Thakar, A. Vahdat, and R. Doyle. Managing energy and server resources in hosting centers. *18th ACM symposium on Operating systems principles (SIGOPS)*, 35(5):103–116, 2001.

[10] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam. Managing server energy and operational costs in hosting centers. *ACM SIGMETRICS Performance Evaluation Review*, 33(1):303–314, 2005.

[11] B.-G. Chun, G. Iannaccone, G. Iannaccone, R. Katz, L. Gunho, and L. Niccolini. An Energy Case for Hybrid Datacenters. *HotPower*, 10 2009.

[12] T. Cortes, C. Franke, Y. Jégou, T. Kielmann, D. Laforenza, B. Matthews, C. Morin, L. P. Prieto, and A. Reinefeld. XtreemOS: a Vision for a Grid Operating System, 2008.

[13] T. Cortes and R. Nou. AEM prototype, D3.3.6, XtreemOS deliverable, 2008.

[14] R. Das, G. Tesauro, J. O. Kephart, D. W. Levine, C. Lefurgy, and H. Chan. Autonomic multi-agent management of power and performance in data centers, 2008.

[15] D. Filani, J. He, S. Gao, M. Rajappa, A. Kumar, R. Shah, and R. Nagappan. Dynamic Data Center Power Management: Trends, Issues and Solutions. *Intel Technology Journal*, 2008.

[16] I. Goiri, J. Guitart, and J. Torres. Elastic Management of Tasks in Virtualized Environments. In *Proccedings of the XX Jornadas de Paralelismo 2009*, pages 671–676, 2009.

[17] V. Hamscher, U. Schwiegelshohn, A. Streit, and R. Yahyapour. Evaluation of job-scheduling strategies for grid computing. *Lecture Notes in Computer Science*, pages 191–202, 2000.

[18] J. Kephart, H. Chan, R. Das, D. Levine, G. Tesauro, F. Rawson, and C. Lefurgy. Coordinating multiple autonomic managers to achieve specified power-performance tradeoffs. In *Autonomic Computing, 2007. ICAC'07.*, pages 24–24, 2007.

[19] B. Khargharia, S. Hariri, and M. Yousif. Autonomic power and performance management for computing systems. *Cluster Computing*, 11(2):167–181, 2008.

[20] C. Lefurgy, K. Rajamani, F. Rawson, W. Felter, M. Kistler, and T. Keller. Energy Management for Commercial Servers. *Computer*, 36(12):39–48, 2003.

[21] L. Liu, H. Wang, X. Liu, X. Jin, W. He, Q. Wang, and Y. Chen. GreenCloud: a new architecture for green data center. In *6th international conference industry session on Autonomic computing and communications*, pages 29–38. ACM New York, NY, USA, 2009.

[22] R. Nou. Energy Efficiency: A case study. Technical Report UPC-DAC-RR-CAP-2009-14, Technical University of Catalonia (UPC) - Computer Architecture Department, 2009.

[23] R. Nou, S. Kounev, F. Julià, and J. Torres. Autonomic QoS control in enterprise Grid environments using online simulation. *J. Syst. Softw.*, 82(3):486–502, 2009.

[24] V. Petrucci, O. Loques, and D. Mossé. A dynamic configuration model for power-efficient virtualized server clusters. In *11th Brazillian Workshop on Real-Time and Embedded Systems (WTR)*, 2009.

[25] V. Petrucci, O. Loques, B. Niteroi, and D. Mossé. Dynamic configuration support for power-aware virtualized server clusters. In *WiP Session of the 21th Euromicro Conference on Real-Time Systems. Dublin, Ireland*, 2009.

[26] E. Pinheiro, R. Bianchini, E. Carrera, and T. Heath. Load balancing and unbalancing for power and performance in cluster-based systems. In *Workshop on Compilers and Operating Systems for Low Power*, volume 180, pages 182–195. Citeseer, 2001.

[27] S. Ranjan, J. Rolia, H. Fu, and E. Knightly. Qos-driven server migration for internet data centers. In *10th International Workshop on Quality of Service (IWQoS 2002)*, pages 3–12. Citeseer, 2002.

[28] S. Rivoire, M. Shah, P. Ranganathan, and C. Kozyrakis. JouleSort: a balanced energy-efficiency benchmark. In *2007 ACM SIGMOD international conference on Management of data*, page 376, 2007.

[29] K. Shen, H. Tang, T. Yang, and L. Chu. Integrated resource management for cluster-based internet services. *SIGOPS OS Rev.*, 36(SI):225–238, 2002.

[30] G. Tesauro. Reinforcement learning in autonomic computing: A manifesto and case studies. *IEEE Internet Computing*, 11(1):22–30, 2007.

[31] G. Tesauro, R. Das, H. Chan, J. Kephart, D. Levine, F. Rawson, and C. Lefurgy. Managing power consumption and performance of computing systems using reinforcement learning. *Advances in Neural Information Processing Systems*, 20, 2007.

[32] D. Vengerov. A reinforcement learning approach to dynamic resource allocation. *Eng. Appl. Artif. Intell.*, 20(3):383–390, 2007.

[33] A. Verma, P. Ahuja, and A. Neogi. Power-aware dynamic placement of hpc applications. In *ICS '08: International Conference on Supercomputing*, pages 175–184, New York, NY, USA, 2008. ACM.

[34] A. Verma, G. Dasgupta, T. Kumar, N. Pradipta, and D. R. Kothari. Server workload analysis for power minimization using consolidation, 2009.

[35] I. H. Witten and E. Frank. Data mining: practical machine learning tools and techniques with java implementations. *SIGMOD Rec.*, 31(1):76–77, 2002.