Server Operational Cost Optimization for Cloud Computing Service Providers over a Time Horizon

Haiyang Qian* and Deep Medhi*
University of Missouri–Kansas City, Kansas City, MO, USA

Abstract

Service providers are migrating to on-demand cloud computing services to unburden the task of managing infrastructure, while cloud computing providers expand the number of servers in their data centers because of the increase in load. With this growing need, their energy consumption increases significantly. Conserving energy and reducing the operational cost while satisfying the service level agreement (SLA) becomes important in order to reduce both carbon emissions and the budget for cloud computing providers. On the other hand, the aggregated demands for different services are dynamic over a time horizon. We present a multi-time period optimization model for saving the operational cost by combining two factors: 1)Dynamic Voltage/Frequency Scaling (DVFS), 2)turning servers on/off over a time horizon. We show the impact of the granularity of the duration of the time slots and frequency options on optimal solutions. A parametric study on varying cost of turning servers on/off and power consumption is also presented.

1 Introduction

Cloud computing service providers (CSPs) (i.e., Amazon EC2, Rackspace Cloudservers™) provide infrastructure on demand and charge based on usage to give flexibility to customers. In this way, just like consuming any other utilities (i.e., water, power, or gas), customers only need to pay for what they have used. CSPs provide infrastructure for service providers (SPs); this service is also known as *Infrastructure as a service (IaaS)*. SPs build their services (*e.g.*, application hosting, content delivery, on-demand work force, search engine, and so on) on top of the infrastructure and offer these services to their end customers. Note that CSPs and SPs do not have to be different entities. Virtualization, such as Linux VServer, VMware, and Xen, are the technologies that enable different services to run in a virtually isolated environment and allow resources that are allocated to these services to scale up and down transparently and seamlessly. These benefits of cloud computing attract more SPs to migrate to the cloud. CSPs expand their data center capacities and/or build more data centers to accommodate this trend.

With the increase in demand, the consumption of power in CSP data centers has increased 400% over the past decade [7]. Even worse, data centers' carbon emissions continue to increase at a speed that is faster than any others. The hard disk is the most vulnerable part in such an infrastructure – majority (78%) of hardware failure/replacement is due to hard disks [14]. Thus, it is important to consider the wear-and-tear cost of hard disks along with the power consumption of servers in order for CSPs to reduce the operational cost.

It has been found that there is significant power consumption when the CPU is idle, i.e., at "base power" [12]. The base power cannot be reduced unless unused hosts are powered off. Thus, an intuitive way to save power is to turn idle servers off. CPU utilization is a good indicator of power consumption because the I/O and memory activities are correlated to CPU utilization and power consumption is a monotonically increasing function with regard to CPU utilization. Many processors have the capability of DVFS, which allows processors to scale the frequency up or down as needed. The cubic relationship between the power consumption and frequency is commonly used, i.e., Power Consumption $= P_{fixed} + P_f \times (\text{frequency})^3$; see [6]. This paper combines these two methods to minimize the server energy consumption and the wear-and-tear cost while satisfying the SLA.

Assuming SPs run CPU intensive services, the demand on the CPU becomes the bottleneck among all server resources. Essentially, CSPs provide resources to their customers (SPs), which usually have non-stationary resource requirements over a time horizon. Therefore,

*This work is supported in part by NSF Grant No. 09-16505.

the demand of SPs is dynamic over a time horizon and needs to be satisfied to a certain degree all the time, including the spike time based on the SLA. The dynamics of demand may cause utilization of CSP data centers to be low if the resources are not optimally assigned. The CSPs' desire is to be able to tune resources based on the demand and required satisfactory level over a time horizon; such changes are to be addressed in a way so that the resources do not remain idle and/or the wear-and-tear cost does not become high due to frequent changes. To understand this issue, we partition the time horizon (period) into multiple time slots. The beginning of each slot is referred to as the review point and assumed to be known in this paper. We define the demand as the sum of new arrivals and previous arrivals still in service and we assume this demand profile is forecasted.

In this paper, we present an optimization model over a time horizon by consider it as a multi-time period problem with demand changing over time, where we address optimizing the operational cost based on both turning on/off and DVFS methods in CSP data centers. Our model is not dependent on the exact duration of a time slot over the time horizon; instead, our model allows us to consider changing the duration of the time slot to understand the impact on the operational cost (while keeping the time horizon fixed). By optimizing the operational cost over the whole time horizon, our model captures the effects caused by demand irregularity and turning on/off cost. We show the impact of granularity of the slot size and CPU frequency options on optimal solutions. The dependence of the optimal slot granularity on cost parameters sheds a light on how to choose optimal slot granularity. We show that it is indispensable to put the problem in the multi-time period framework. Although we use hypothetical demand distribution in our numeric study, our approach is applicable to arbitrary demand distribution.

The remainder of this paper is organized as follows. We present the optimization formulation in Section 2. In Section 3, the evaluation environment setup and results are presented. Section 4 summarizes the related work. Section 5 discusses conclusion and further work.

2 Problem Formulation

We first introduce our notations. The CSP data center has I servers. Let $\mathcal{I}$ denote the set of the servers. Let $\mathcal{J}(i)$ be the frequency option set for server i. In a homogeneous server cluster case, $\mathcal{J}(i) = \mathcal{J}, \forall i$. There are J frequency options in $\mathcal{J}$. Server i running at j-th frequency option can offer a capacity of V_{ij} while satisfying the SLA. The power consumption of running server i at j-th frequency is denoted by C_{ij} per time unit. The wear-and-tear cost of turning a server on and off is denoted by C_s^+, C_s^-, respectively. Let $\mathcal{K}$ be the set of the services that is using the cloud computing infrastructure consisting of K services. We divide the time horizon Υ hours into T equal time slots and the *duration of a time slot (slot size)* is then $\tau = \Upsilon/T$ hours (usually, we refer to the slot size in minutes). Let $\mathcal{T}$ be the set of these slots. At the review point (starting point of each time slot), the number of active servers and their frequencies is configured. Let binary decision variables $y_{ij}(t)$ denote if server i is running at frequency option j at time slot t. The continuous variables $x_{ijk}(t)$ ($0 \leq x_{ijk}(t) \leq 1$) represent the proportion of service k hosted by server i that is running at frequency option j at time slot t.

There are a number of constraints in this problem. Usually, a server can only be operated at a specific frequency in a time slot—this can be represented as follows:

$$\sum_{j\in\mathcal{J}} y_{ij}(t) \leq 1, \ \forall i \in \mathcal{I}, \forall t \in \mathcal{T}. \tag{1}$$

The second constraint is that the total utilization of any running server must not be more than 1 and that of any "off" server must be 0.

$$\textstyle\sum_{k\in\mathcal{K}} x_{ijk}(t) \leq y_{ij}(t), \ \forall i \in \mathcal{I}, \ \forall j \in \mathcal{J}, \ \forall t \in \mathcal{T}. \tag{2}$$

The third constraint is the demand requirement of each service over the time horizon:

$$\textstyle\sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} V_{ij} \cdot x_{ijk}(t) \geq D_k(t), \ \forall k \in \mathcal{K}, \ \forall t \in \mathcal{T}. \tag{3}$$

The constraint (3) is a combined constraint of assigning servers to services and satisfying the demand of each service. We do not consider the assignment/allocation problem in this paper although the service assignment problem is an important one [10]. Enabled by virtualization and consolidation, (2) and (3) can be combined as the aggregated demand constraint:

$$\sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} V_{ij} \cdot y_{ij}(t) \geq \sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} V_{ij} \cdot x_{ijk}(t) \geq \sum_{k\in\mathcal{K}} D_k(t). \tag{4}$$

Let $\sum_{k\in\mathcal{K}} D_k(t) = D(t)$; then, the demand constraint reduces to:

$$\textstyle\sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} V_{ij} \cdot y_{ij}(t) \geq D(t), \ \forall i \in \mathcal{I}, \forall t \in \mathcal{T}. \tag{5}$$

We next consider the objection function. The objective is to minimize the operational cost of running servers over the entire horizon that can be represented by

$$\sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} C_{ij} \cdot y_{ij}(t) + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} \Big(C_s^+ \cdot \sum_{j\in\mathcal{J}} y_{ij}(t) \cdot \Big(\sum_{j\in\mathcal{J}} y_{ij}(t) - \sum_{j\in\mathcal{J}} y_{ij}(t-1) \Big) + C_s^- \cdot \sum_{j\in\mathcal{J}} y_{ij}(t-1) \cdot \Big(\sum_{j\in\mathcal{J}} y_{ij}(t-1) - \sum_{j\in\mathcal{J}} y_{ij}(t) \Big) \Big).$$

This objective function is a quadratic function. Given that $y_{ij}(t)$ are decision variables, we can reduce the quadratic function to a linear function (by introducing additional variables and constraints) without resorting to any approximation. To do this, we introduce two binary variables $y_i^+(t)$ and $y_i^-(t)$ to represent turning on/off at review point of time slot t. $y_i^+(t) = 1$ means server i is turned on at time t. Conversely, $y_i^-(t) = 1$ means server i is turned off at time t. 0 indicates no change from time slot $t-1$ to t. Thus, we have $\forall i \in \mathcal{I}, \forall t \in \mathcal{T}$,

$$\sum_{j \in \mathcal{J}} y_{ij}(t) - \sum_{j \in \mathcal{J}} y_{ij}(t-1) - y_i^+(t) + y_i^-(t) = 0. \quad (6)$$

Since $y_i^+(t)$ and $y_i^-(t)$ cannot be both 1 at the same review point, we use the following inequalities to force this requirement:

$$y_i^+(t) + y_i^-(t) \leq 1, \ \forall i \in \mathcal{I}, \forall t \in \mathcal{T}. \quad (7)$$

With the aid of $y_i^+(t)$ and $y_i^-(t)$, the original quadratic objective function can be transformed to the following linear function:

$$\sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} C_{ij} y_{ij}(t) + \sum_{t \in \mathcal{T}} \sum_{i \in \mathcal{I}} \Big(C^+ y_i^+(t) + C^- y_i^-(t) \Big). \quad (8)$$

Since we consider the planning horizon to be the entire time period Υ, we make the assumption that at the beginning of this period, a reshuffling is done. In other words, all servers are reset at the beginning of the time horizon. In our case, we use $y_{ij}(0) = 0, \forall i \in \mathcal{I}, \forall j \in \mathcal{J}$, while other values may be used as initialized values. This then forces the new binary variables in the following way:

$$y_i^+(1) = \sum_{j \in J} y_{ij}(1), \ \forall i \in I, \quad y_i^-(1) = 0, \ \forall i \in I. \quad (9)$$

To summarize, the optimization problem is to minimize the objective function (8) that is subject to (1), (5), (6), (7), and (9), with all variables being binary.

3 Evaluation

In this section, using the optimization model, we focus on understanding: 1) the impact of the granularity of time slot size on optimal solutions, 2) to what extent DVFS can improve the optimum, and 3) how the relative change in the wear-and-tear cost compared to the power consumption cost impact on the optimal solution. In order to do these studies, we start with our evaluation setup and parameters values considered.

3.1 Evaluation Setup

In our study, we consider a server cluster of 100 identical servers. The CPU frequency set and power consumptions are adopted from [6]. The capacity of a server (V_j') is assumed to be a linear function of frequencies (F_j) with a fixed cost that is calculated as follows:

$$V_j' = \alpha + F_j, \forall j \in \mathcal{J}, \text{ where } \alpha \in \mathbb{R}, \alpha > -F_j. \quad (10)$$

There are different units that can be used for server capacity such as processors requested, HP computon, or SAPS. Without loss of generality, we can simply use numeric values to convey this information while assuming that the demand is using the same unit. We capitalize on α to make our model represent different measurements. For ease of computation, we normalize the capacity at j to be the capacity at the maximum frequency. That is,

$$V_j = V_j'/(\alpha + F_J). \quad (11)$$

Note that to keep the problem consistent, we scale the demand by the same normalization factor:

$$D(t) = D'(t)/(\alpha + F_J). \quad (12)$$

In [8], Greenberg *et. al.* used \$.07 per KWH as the utility price. We use the same utility price. The operational cost in a time slot is the product of the power consumption, the utility price and the slot size. Google reported [1] that the personnel cost for each repair is \$100 and the replacement cost is 10% of the server cost (\$2000). We assume the lifetime for a disk to be 60,000 turning-on-and-off cycles. Using this, we arrive at 0.5 cents for the wear-and-tear cost per turning-on-off cycle. Out of 0.5 cents, we assume turn on to be a higher cost than turning off; thus, we split this value to 0.3 cents and 0.2 cents for turning on and turning off, respectively. We also assume that the power cost for turning on and turning off is 0.02 cents and 0.005 cents, respectively, since turning on draws much more power than turning off in most cases. This analysis of the turning on/off cost is similar to the one in [6] except that we differentiate the cost of turning on and off. We summarize the cost parameters in Table 1. The normalized capacity shown in Table 1 is an instance based on (11) given $\alpha = 0$ that makes the relative difference of capacities for different frequencies the largest. Therefore, this is the case that yields the upper limit of the the benefits that DVFS can contribute.

Since the maximum capacity per server is normalized to 1, the maximum cluster capacity is equal to the number of servers. We assume that the demand profile is forecasted and profiled every 5 minutes based on traces of demand on the CPU. Due to the diurnal behavior associated with human beings' working cycles, we chose the 8 hour work time as the planning horizon where the dynamically changing demand from one time slot to another is generated for our study. We consider 5 different time slot sizes: 5 minutes, 15 minutes, 30 minutes, and 60 minutes. Based on the 5 minute demand profile, the demand for larger time slot granularity is taken to be the

Frequency Option	j	1	2	3	4	5	6	7	8
Frequency (GHz)	F_j	1.4	1.57	1.74	1.91	2.08	2.25	2.42	2.6
Normalized Capacity	V_j	.5385	.6038	.6692	.7346	.8	.8645	.9308	1
Power Consumption (watts)	P_j	60	63	66.8	71.3	76.8	83.2	90.7	100
Power Cost (cents)	C_j	$.42\tau$	$.441\tau$	$.4676\tau$	$.4991\tau$	$.5376\tau$	$.5824\tau$	$.6349\tau$	$.7\tau$

Table 1: **CPU frequencies, capacities and operational cost**

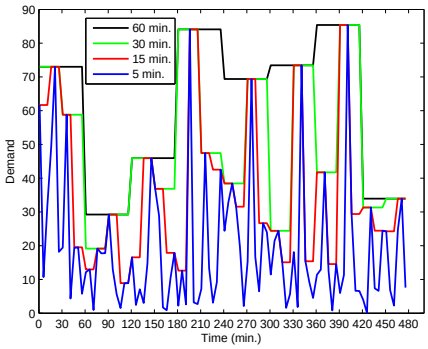

Figure 1: **Demand profile for different time slot granularity**

maximum over all 5-minute demands in that slot:

$$D(t, t+\Delta) = \max\{D(t), \cdots, D(t+\Delta)\} \qquad (13)$$

where $D(t, t+\Delta)$ denotes the maximum demand of Δ 5-minute slots (from t to $t+\Delta$). Fig. 5 shows the demand profile for different time slot granularity in the 100 server case. As we can see, the demand profile of the coarse grain time slot is to envelop the demand profile of the fine grain time slot.

To study the benefits of DVFS on the optimal solutions and provide some insights for the CPU frequent management, we evaluate three schemes: 1) The CPU does not have DVFS capability and always runs at maximum frequency; this scheme serves as the baseline study and will be denoted by "*Max*"; 2) The CPU can scale only at the minimum and the maximum frequencies — this is denoted by "*PingPong*"; 3) The CPU can scale in the full spectrum of 8 frequency options — this is denoted by "*Full*". To show the benefits of our approach, we also present two baseline cases: 1) All servers are always on and run at maximum frequency, that is, no optimization and cost management is employed — this is referred to as "*Baseline-I*", 2) Optimization is done independently at each time slot — this is referred to as "*Baseline-II*".

We ran the optimization model using CPLEX through its integer programming solver on an Intel(R) Pentium(R)IV 3.00GHz with 2GB memory. Through preliminary runs, we observed that the overall cost does not improve if we allow 2,000 branch and cut nodes in CPLEX; thus, in our study, we set the branch and cut nodes limit to 2,000.

3.2 Results and Discussions

Fig. 2 depicts the minimum cost in a 100-server cluster with 20% utilization where the demand distribution is assumed to be exponential. The upper three curves show the minimum cost when the turning on/off cost is considered while the lower three curves show the minimum cost when the turning on/off cost is *not* considered.

We start our discussion for the case that assumes no turn on/off cost, i.e., when $C_s^+ = C_s^- = 0$. From Fig. 2, we note that the granularity of time slot size has significant impact on the optimum. By increasing the time slot size from 5 minutes to 60 minutes, the minimum cost increases by more than 200% for all three schemes. This is not surprising since (13) results in over-provisioning.

In the case when the turning on/off cost is considered, the Full scheme results in the minimum cost but only slightly less than the PingPong scheme while both of these schemes outperform the Max scheme significantly. The intrinsic reason for performing optimization periodically, instead of continuously, is the overhead caused by optimization and resulting operations. This overhead of DVFS is much smaller than that of turning on/off.

Consider next the relative differences between the case with the turning on/off cost and the case without. The granularity of time slot size has a similar impact on optimum as the case that assumes no turning on/off cost. But the increase in the slope of the minimum cost is much less than in the case with no turning on/off cost. By taking this factor into consideration, finer granularity of the time slot size requires more frequent optimization operations, which cause more turning on/off operations. This cost cancels out part of the savings brought by finer granularity of the slot size. Essentially, our approach intends to seek an optimal operating point for the accurate provisioning to the demand profile to reduce the power consumption cost and to minimize turning on/off operations (to reduce the wear-and-tear cost).

Fig. 2 also includes the optimal solutions for Baseline-I and Baseline-II. Note that the turning on/off cost is considered in these two baseline cases. Recall that, in Baseline-I, the configuration is static where servers are always operated at the maximum frequency and only the turning on cost incurs at the beginning of the first time slot. We define the ratio of relative difference between Baseline-I and our approach, considering turning on/off cost, as the *relative improvement*. As shown in Fig. 3,

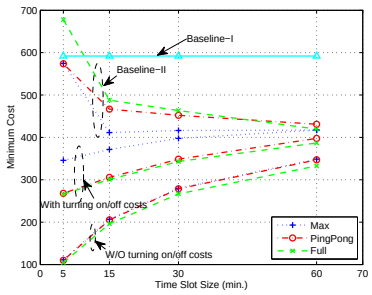

Figure 2: **Minimum cost in a 100 server cluster**

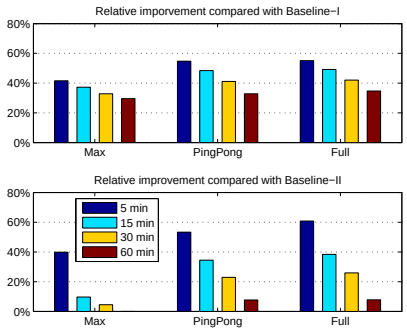

Figure 3: **Relative improvement**

the relative improvement is significant; clearly, the finer granularity of the time slot size, more than the frequency options, helps to reduce the minimum. In Baseline-II, optimization is done independently for each time slot. In the presence of the turning on/off cost, this local optimization can mislead the global optimum. As we can see from Fig. 2, in the case of 5 minute time slot size using Full scheme, the optimal solution obtained in Baseline-II is even worse than Baseline-I. When the time slot size is small, Baseline-II minimizes the server power consumption in each time slot independently and causes too many turning on/off operations that outweigh the savings in power consumption. The relative improvement of our approach compared with Baseline-II is shown in Fig. 3. As the time (slot size) granularity becomes larger, the improvement diminishes gradually. For large time granularity, the frequency of turning on/off operations is low; thus, the difference of optimizing for each time slot independently and multiple time slots considered together is small. This also explains that the optimum becomes lower as the time slot size granularity becomes larger, as shown in Fig. 2 as the turning on/off cost in Baseline-II cancels out the gain of the savings in the power cost by finer time slot size granularity. In essence, this also shows the value of the multi-time period framework.

The relative difference between the cost of turning on/off and the server power consumption cost also can have a different impact. To see this, we scale the turning

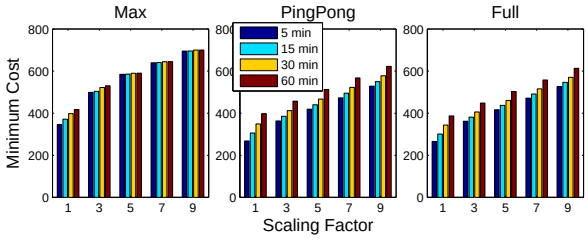

Figure 4: **Scaling Factor Versus Minimum cost**

on/off cost by factor R and study the impacts of relative difference in cost. Fig. 4 presents the optimal solutions for the cases whose turning on/off cost is scaled from 1 to 9 compared to the server power consumption cost. As we can see, the discrepancy of different granularity of time slot size becomes smaller along with the scaling factor increasing from 1 to 9. This trend is consistent for all 3 frequency schemes. Depending on the overhead, finer granularity does *not* always gives the optimal solution. If the turning on/off cost is negligible compared with the energy consumptions for all frequencies, this case can be regarded as equivalent to the case without considering the turning on/off cost; this would suggest that the finer granularity of time slot size is the best option to do. However, if the power consumption cost is negligible, we may avoid turning on and off operations and, instead, choose coarse granularity of time slot size.

4 Related Work

A significant amount of work has addressed energy savings and operational cost reduction in server cluster environments [2–6, 11, 12]. Most of these works evaluated their approaches based on a relatively small server cluster size (around 10). While some have considered a cluster size of 100 (same as our work), they do not address the time horizon.

Turning the server on and off as needed was originally proposed by Pinheiro *et. al.* [12] to save energy; they used the Proportional-Integral-Differential (PID) method based on *control theory* to predict the demand for the next decision point. This method takes the current demand status, previous accumulated demand status, and demand change speed into consideration and gives different weights to decide the demand for the next decision point. Bichler *et. al.* used a mixed-integer programming model to formulate the capacity planning problems for virtualized servers [4]. They also identified the "static server allocation problem" as an instance of the bin packing problem. However, DVFS is not considered in their work, nor the cost dependency over the time horizon due to the on/off cost.

Petrucci *et. al.* considered both turning on/off and DVFS to formulate the "virtual server cluster configuration problem" by a mixed integer programming model

and presented an algorithm to dynamically manage server clusters [11]. They proposed to devise a control loop to periodically run the optimization problem to adapt to the time-varying incoming workload of multiple applications, and formulated the problem as a single time period problem and thus, can only take the turning on/off cost based on adjacent time periods into consideration, which results in a local optimum. Chen *et. al.* [6] formulated the objective function considering power consumption and the turning on cost (without the turn off cost) for a different types of constraints than the ones we identified; in addition, they predicted the first and second moment of the next interval arrivals and finally calculated the SLA constraint in terms of delay based on the $G/G/m_i$ queue. However, the effects of time slot granularity of this multi-time period problem on the optimal cost is not discussed in their work.

Multi-time period problems have been studied over three decades in related areas such as transportation research, inventory management, and telecommunication network design [9, 13]. While in many of these problems, the dependency arises in the form of constraints due to the remaining capacity of one period being used in a subsequent period, the dependency in our case is primarily in the form of the turning on/off cost.

5 Conclusion and Further work

In this work, we use the wear-and-tear cost and power consumption to capture the server operational cost in CSP data centers. The demand is considered to be dynamically changing over a time horizon. CSPs desire to offer capacity just as needed to avoid over-provisioning; however, capacity re-assignment incurs the wear-and-tear cost. Leveraging two well-known methods, turning servers on/off and DVFS in a synchronous manner, we presented a multi-time period mathematical programming model to optimize the operational cost in this cloud computing environment. The evaluation study shows that our approach can significantly reduce the server operational cost compared with static capacity allocation (baseline-I) and when optimized locally (baseline-II); we found that the time slot size granularity has pronounced effects on the optimal solutions. To make our cost assumptions general, we study how the optimal time slot granularity is impacted by the relationship between two kinds of costs.

Our work is still at a preliminary stage. One of our next steps is to develop a heuristic algorithm to handle large cluster sizes (such as when 1,000 or 10,000 servers). In addition, there are a number of factors we have not incorporated so far in our work. Consider, for instance, turning servers on/off; a cluster level management method incurs more management operations during the turning on/off processes than DVFS does during frequency scaling processes. Our model does not consider this overhead. Secondly, the intrinsic reason for performing optimization periodically, instead of continuously, is the overhead caused by optimization and resulting operations. This overhead of DVFS is much smaller than that of turning on/off. In our current model, the two methods operated in a synchronous manner. It would be worthwhile to consider a model that is able to asynchronously use these two methods. Thirdly, we consider the problem for a known set of forecasted demand over a time horizon. A model that addresses uncertainty in demand over the time horizon would reflect a more realistic situation. These aspects will be considered in the future.

References

[1] BARROSO, L. A., AND HÖLZLE, U. *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines*. Morgan and Claypool Publishers, 2009.

[2] BERTINI, L., LEITE, J. C. B., AND MOSSÉ, D. Power optimization for dynamic configuration in heterogeneous web server clusters. *J. Syst. Softw. 83*, 4 (2010), 585–598.

[3] BIANCHINI, R., AND RAJAMONY, R. Power and energy management for server systems. *IEEE Computer 37* (2004), 2004.

[4] BICHLER, M., SETZER, T., AND SPEITKAMP, B. Capacity planning for virtualized servers. In *Workshop on Information Technologies and Systems (WITS)* (Milwaukee, Wisconsin, 2006).

[5] BOHRER, P., ELNOZAHY, E. N., KELLER, T., KISTLER, M., LEFURGY, C., MCDOWELL, C., AND RAJAMONY, R. *The case for power management in web servers*. Kluwer Academic Publishers, Norwell, MA, USA, 2002, pp. 261–289.

[6] CHEN, Y., DAS, A., QIN, W., SIVASUBRAMANIAM, A., WANG, Q., AND GAUTAM, N. Managing server energy and operational costs in hosting centers. *SIGMETRICS Perform. Eval. Rev. 33*, 1 (2005), 303–314.

[7] FILANI, D., HE, J., GAO, S., RAJAPPA, M., KUMAR, A., SHAH, R., AND NAAPPAN, R. Dynamic data center power management: Trends, issues and solutions. *Intel Technology Journal* (2008).

[8] GREENBERG, A., HAMILTON, J., MALTZ, D. A., AND PATEL, P. The cost of a cloud: research problems in data center networks. *SIGCOMM Comput. Commun. Rev. 39*, 1 (2009), 68–73.

[9] JOHNSON, L. A., AND MONTGOMERY, D. C. *Operations Research in Production Planning, Scheduling, and Inventory Control*. John Wiley & Sons, 1974.

[10] MENG, X., PAPAS, V., AND ZHANG, L. Improving the scalability of data center networks with traffic-aware virtual machine placement. In *INFOCOM* (2010).

[11] PETRUCCI, V., LOQUES, O., AND MOSSÉ, D. Dynamic optimization of power and performance for virtualized server clusters, Technical Report, 2009.

[12] PINHERIO, E., BIANCHINI, R., CARRERA, E. V., AND HEATH, T. Dynamic cluster reconfiguration for power and performance. In *Compilers and Operating Systems for Low Power* (2003), L. Benini, M. Kandemir, and J. Rammanujam, Eds., Kluwer.

[13] PIÓRO, M., AND MEDHI, D. *Routing, Flow, and Capacity Design in Communication and Computer Networks*. Morgan Kaufmann Publishers, 2004.

[14] VISHWANATH, K. V., AND NAGAPPAN, N. Characterizing cloud computing hardware reliability. In *Proc. of 1st ACM Symposium on Cloud Computing* (June 2010).