Data Center Resource Management with Temporal Dynamic Workload

Haiyang Qian* and Deep Medhi
University of Missouri–Kansas City, USA
{hqian,dmedhi}@umkc.edu

***Abstract*—The proliferation of Internet services drives the data center expansion in both size and the number. More importantly, the energy consumption (as part of the total cost of ownership (TCO)) has become a social concern. When the workload demand is given, the data center operators desire minimizing their TCO. On the other hand, when the workload demand is unknown while the requirements on quality of experience (QoE) of the Internet services are given, the data center operators need to determine the appropriate amount of resources and design redirection strategies in presence of multiple data centers to guarantee the QoE.**

For the first problem, we present formulations to minimize server energy consumption and server cost with dynamic temporal demand and propose novel aggregation methods to reduce computational complexity. The Dynamic Voltage/Frequency Scaling (DVFS) capacity is further considered in our model. Our numerical results show that adopting DVFS results in a significant reduction of energy consumption. For the second problem, the data center provides resources via the cloud computing model. We propose a hierarchical modeling approach that can easily combine all components in the data center provisioning environment. The numeric results show that our model serves as a very useful analytical tool for data center operators to provide appropriate resources as well as design redirection strategies.

I. Introduction

The cost of data centers has become one of major social and economic concerns. It has been reported that the power consumption in data centers has increased 400% over the past decade [21]. Data centers consumed 61 billion kilowatt-hours of power in 2006, according to a report of the U.S. Environmental Protection Agency (EPA) in 2007 [44]. That is 1.5 percent of all power consumed in the United States–at a cost of 4.5 billion U.S. dollars. Moreover, data center energy costs are approaching overall hardware costs [4]. The carbon footprint of data centers contributes to global warming. People become more and more aware of the needs to "go green" for data centers. Reducing energy consumption in data centers is an extremely high concern.

There are infrastructure expenditures (CAPEX) and operational expenditures (OPEX) in a data center. The energy consumption belongs to OPEX. The energy is consumed by multiple units in data centers, such as servers, cooling and power distribution loss. In this dissertation, we focus on minimizing the costs of servers, i.e., energy consumption to run servers and the CAPEX of servers.

The popular pay-as-you-go model adopted by many cloud computing providers (*e.g.*, Amazon EC2 [1], Rackspace Cloudservers™) is one of the most attractive features for customers whose demands on resources vary over time. Just like consuming any other utilities (*e.g.*, water, gas), customers pay for what they have consumed. Cloud computing providers use data centers as underlying infrastructures and provision resources to Internet services [3], [23], [48].

The real-world data center workload trace reveals that the workload is highly dynamic [17], [5], [32], [2], [15]. The workload fluctuates over timescales of minutes, hours and days [17]. There are many works on predicting the workload in data center environments, *e.g.*, [22], [45]. Thus, we assume the workload is predictable. In this dissertation, the workload demand is given at a 5-minute interval. A 5-minute interval is called a time slot. The beginning of each slot is referred to as the *review point*. We define the workload at a certain time slot as the total amount of resources required by both new arriving jobs at the current review points slot and previous jobs that are still active.

The data center operators have to provision enough resources to satisfy the peak workload demand. Statically provisioning results in over-provisioning for off-peak workload demands. The power needed at peak workload is called peak power. It has been found that there is significant power consumption when the processor is idle (*i.e.,* it is at "base power" [32]). It is measured that idle servers consume more than 66% of the peak power [17], [20]. The base power cannot be reduced unless unused hosts are powered off. Processor (CPU) utilization can be used as an indicator of power consumption because the I/O and memory activities are correlated to processor utilization and power consumption is a monotonically increasing function with regard to processor utilization [11]. 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 [18], [19]. This relationship is given by

$$\text{Power Consumption} = P_{fixed} + P_f \times (\text{frequency})^3, \quad (1)$$

where P_{fixed} is the fixed component and P_f is the coefficient of the frequency related component.

Virtualization is the technology that enables different services to run in a virtually isolated environment and allows resources that are allocated to these services to scale up and down transparently and seamlessly [14]. Cloud computing providers use one virtual machine (VM) (It is called an instance in Amazon EC2) as the finest granularity of resources to offer the service. Consolidation is the technology that utilizes the capability of migration to consolidate the workload into a minimum number of running servers and maximize the number of idle servers

*Haiyang Qian is now with China Mobile USA Research Center, Milpitas CA, USA. Email: haiyangqian@chinamobile.com. This project is supported in part by NSF Grant CT-08-31090, CNS-09-16505 and PhoneFactor Inc. (acquired by Microsoft in Oct., 2012).

978-3-901882-50-0 ©2013 IFIP

("Minimum number of running servers" does not refer to fully utilize the server since we may not fully utilize servers on purpose to guarantee Quality of Service (QoS).) [43].The way of assigning servers is called workload dispatching. Assigning resource (servers) in the Last-Released-First-Used (LRFU) manner can minimize the needs of migration for consolidation [28]. That is, we use the server that is released most recently when the workload increases. We assume LRFU job dispatching in this dissertation.

The above facts suggest that data center operators consolidate jobs into a minimum number of servers and switch idle servers off. However, switching servers on/off results in wear-and-tear costs and consolidation costs. The hard disk is the most vulnerable part in data center infrastructure – majority (78%) of hardware failure/replacement is due to hard disks [46]. To represent the "wear-and-tear" costs of switching on and off, we amortize the CAPEX of servers by dividing the average price of the servers by the average number of switching on/off cycles of the hard disk. In addition, the source server needs to run for additional time to keep the states of the running application when applying consolidation. This additional time consumes extra energy. The presence of relationships between the states of adjacent time slots requires a global optimization framework in the entire planning horizon. We use integer programming to determine the optimal number of running servers at review points such that the costs of energy consumption and CAPEX of servers are minimized in the entire planning horizon. The optimal solution should be computed in an approximately real-time manner, even for large scale data centers.

The users are served by the Internet service providers (ISeP) who in turn, are the customers of the cloud computing service provider (CSP) [3]. And CSP use data centers as their infrastructure. *QoE* is critical for service providers since their revenue depends on it [25]. Being able to receive service and the time needed to wait comprises QoE for online service users. ISePs desire to be able to predict the QoE of their users. The service performance described in a service level agreement (SLA) includes response time, utilization, throughput, and availability [40], [41], just to name a few [47]. However, these aspects consist of a promised service level that is delivered from the data center to the ISePs. None of these aspects is suitable to be used as a direct barometer of QoE of the online service. Note the response time in a data center, however, does not necessarily correlate to the QoE since the response time experienced by users is the summation of Internet delay and the response time in the data center, and the redirection functionality further complicates the QoE. In addition, it is desirable for ISePs to be able to evaluate the QoE by simple metrics that comprehensively address the service delivered by CSP and underlying data centers, the Internet's conditions, and redirection strategies. Our proposed hierarchical model meets this need.

A. Contributions of this Dissertation

The contributions of this dissertation are as follows. (1)We formulate the data center server operational cost minimization problem in three data center server environments: homogenous, heterogeneous and hybrid hetero-homogeneous data centers when the resource workload requirements are temporally dynamic. Workload demand aggregation is proposed to reduce computational time to solve the problem. Part of this work has been published in [36]. (2)We further take Dynamic voltage/frequency scaling (DVFS) into considering. This work has been published in [38]. (3)We propose a hierarchical modeling approach to evaluate the QoE of users when using internet services hosted by cloud computing providers. This work has been published in [39]. (4)We conduct a study on a specific Internet service, called agent-based VPN and propose redirection strategies to balance the service denial probability and latency. Based on this work, a conference paper was initially published in [33], which was later extended with additional results for a journal paper [35]. (Due to the space limit, this part of work is not introduced in this paper.)

B. A Literature Survey

A significant amount of work focuses on saving energy on servers in data centers [8], [9], [10], [13], [18], [19], [26], [31], [32], [17], [28], [7], initiated by the pioneering work [16], [32] a decade ago. Aiming at reducing data center server operational costs, these works use different techniques, such as mathematical programming, queuing theory, control theory, machine learning to model the problem, and develop algorithms and heuristics. A recent comprehensive survey can be found at [6].

Reducing server energy consumption can be performed at different levels. At the individual physical server level, the capacity of server and its components is scaled according to load. The load of the processor is usually called utilization. As we have mentioned before, processor utilization can be used to indicate the power consumption. Processors with DVFS capacity can scale the voltage/frequency according to load and therefore, reduce the power consumption. Barroso and Hölzle proposed the term *energy proportinality* to refer to the energy consumption proportional to the load [5]. The cost associated with the physical server level is the power consumption. At the cluster level, the decisions are made regarding how many servers are needed. Switching the server on and off as needed was originally proposed by Pinheiro *et. al.* [32] to save energy. The cost associated with the cluster level is the wear-and-tear cost. At the virtualization level, migration and consolidation are used to consolidate the workload into a lesser number of virtual machines. The cost associated with the virtualization level is the consolidation cost. Many works in the literature consider more than one level of factors. [32] uses the Proportional-Intergral-Differential (PID) method based on the control theory to predict the demand for the next decision point. Bichler *et al.* use mixed-integer programming to formulate the capacity planning problems for virtualized servers [10]. However, DVFS is not considered in their work. Petrucci *et. al.* consider both turning on/off and DVFS to formulate the so called "virtual server cluster configuration problem" by mixed integer programming and present an algorithm to dynamically manage server clusters [30], [29], [31]. They formulate the problem in a single time period and thus can only take the turning on/off cost based on the adjacent time periods into consideration, which results in a local optimum. Chen *et al.* first formulate the objective function, then predict the first and second moment of the next interval arrivals and finally calculate the SLA constraint in terms of delay based on $G/G/m_i$ queue [18]. They also propose an approach based on control theory and hybrid approach of queueing and control theory as alternatives. Most of these works evaluated their approaches based on a relatively small server cluster size (such as around 10), while some have considered a cluster size around 100.

How the dynamic workload structure affects the optimal solution and how to use the workload structure to determine the

optimal solution has not been received a lot of attention. [37], [28] decompose the workload into certain substructures and utilize the substructures to determine the optimal solution. Lu and Chen also present a method to solve the problem when the workload is unpredictable [28].

Garfinkel conducted a number of experiments to evaluate the Amazon's Grid Computing services. Li *et al.* benchmarked different cloud services [27]. A number of cloud computing hardware reliability issues are identified for future research in [46]. Kalyanakrishnan *et al.* used collected data to study the host reliability from the perspective of the users [24]. Bodik *et al.* discussed the affect of turning machines on and off on reliability in [12].

C. Organization

The rest of this dissertation paper is organized as follows. Sec. 2 formulates the data center resource allocation problem and presents time slot aggregation methods to reduce the computational complexity when solving the problem. DVFS is considered along with switching servers on and off to formulate the problem in Sec. 3. Sec. 4 introduces the hierarchical model to evaluate Quality of Experience for Internet Services hosted in data centers. Sec. 5 concludes this dissertation paper.

II. Energy-Aware Data Center Resource Management Given Resource Requirements

We formulate models to determine the optimal number of running servers. Workload aggregation methods are proposed to reduce the computational time of computing optimum. Two workload aggregation modes are introduced for different needs of applications. To combat the pitfalls of static aggregations for both modes, we further introduce dynamic aggregations.

A. Problem Formulation

There is a data center or a cluster in a data center with I servers. Let $\mathcal{I}$ denote the set of servers, while the cardinality of this set is denoted by I (i.e., $\#(\mathcal{I}) = I$). This optimization problem is considered in a duration of Υ hours. In practice, we expect Υ to be six to eight hours during which the workload can be accurately predicted according to history observations. There are many techniques can be used to achieve an accurate prediction, *e.g.*, [22], [45]. However, the prediction techniques are out of the scope this paper. The duration Υ hours are divided into T equal *time slots* or *periods*(The terms time slot and period are used interchangeably in this paper.) and the duration of a time slot, called slot size, is $\tau = \Upsilon/T$ hours. We assume that the workload on the CPU is predicted at the beginning of the planning duration. The servers are reconfigurable at the beginning of the time slots. These points are called *review points*. The capacity that server i at time slot t can offer is denoted by v_{it}. We assume the capacity offered by server i is the same for all time slots. Thus, we have

$$v_{it} = v_i,\ t = 1, \cdots, T. \tag{2}$$

We want to determine how to configure the servers at review points such that the total cost of energy consumption and server CAPEX is minimized over the entire planning period.

There are three models: (1) All servers to be heterogenous and this model is denoted by *Model-Het*; (2) All servers are identical and this model is denoted by *Model-Hom*; (3) There are different clusters of servers and each cluster has a certain number of homogeneous servers while servers may be different from one cluster to another. Thus, this is a hybrid of the previous two models. This model can also be used when homogenous servers are required to partition into multiple clusters for ease of management. This is denoted by *Model-HH*. Because the third model is the generalized case of the first two models, we only present the formulation for the Model-HH here.

Denote the set of clusters by $\mathcal{J}$ and its cardinality, $J = \#(\mathcal{J})$, where $1 \leq J \leq I$, represents the number of clusters. When $J = 1$, it is Model-Hom; when $J = I$, it becomes Model-Het. Let the energy consumption of running a server in cluster j at a time slot be z_j^p. Denote the number of servers in cluster j by I_j, where $\sum_j I_j = I$. We denote the set of the number of running servers for cluster j by $\mathcal{N}_j$, which can be $0, 1, \cdots, I_j$. Let z_{jt} be the number of running servers in cluster j at time slot t. Performing workload dispatching is the overhead in this model. To achieve this model, we need to use the resource in the LRFU manner. Implementing workload dispatching algorithms is out of the scope of this paper. It has been studied in [17], [28]. However we need to keep in mind that this overhead happens Model-Hom as well. The cost of running a server in cluster j is denoted by c_j^p. The costs of switching a server in cluster j on and off are represented by c_j^{s+} and c_j^{s-}, respectively. The capacity of a server in cluster j is given by v_j. Another way to look at this problem is to consider it as the superposition of multiple homogeneous cases.

We now introduce constraints in this problem. First, the workload requirements need to be satisfied all the time:

$$\sum_{j \in \mathcal{J}} v_j \cdot z_{jt} \geq d_t,\ t = 1, \cdots, T. \tag{3}$$

Secondly, the number of running servers cannot be larger than the total number of servers in that cluster:

$$z_{jt} \leq I_j,\ j = 1, \cdots, J;\ t = 1, \cdots, T. \tag{4}$$

Let z_{jt}^+ be the number of servers turned on in cluster j at the review point of time slot t while z_{jt}^- be the number of servers turned off in cluster j at the review point of time slot t. Then z_{jt}^+ should take maximum between 0 and $z_{jt} - z_{j(t-1)}$:

$$z_{jt}^+ = \max\{0, z_{jt} - z_{j(t-1)}\},\ j = 1, \cdots, J;\ t = 1, \cdots, T.. \tag{5}$$

Similar to Z_{jt}^+, we have:

$$z_{jt}^- = \max\{0, z_{j(t-1)} - z_{jt}\},\ j = 1, \cdots, J;\ t = 1, \cdots, T.. \tag{6}$$

And the objective function is given by

$$F = \sum_{t \in \mathcal{T}} \sum_{j \in \mathcal{J}} (c_j^p \cdot z_{jt} + c_j^{s+} \cdot z_{jt}^+ + c_j^{s-} \cdot z_{jt}^-) \tag{7}$$

Because this is a minimization problem with the cost coefficients being non-negative, we can substitute (5) by following two linear inequalities:

$$z_{jt}^+ \geq z_{jt} - z_{j(t-1)},\ j = 1, \cdots, J;\ t = 1, \cdots, T. \tag{8}$$

$$z_{jt}^+ \geq 0,\ j = 1, \cdots, J;\ t = 1, \cdots, T. \tag{9}$$

Likewise, we can substitute (6) by following two linear inequalities:

$$z_{jt}^- \geq z_{jt} - z_{j(t-1)},\ j = 1, \cdots, J;\ t = 1, \cdots, T. \tag{10}$$

$$z_{jt}^- \geq 0,\ j = 1, \cdots, J;\ t = 1, \cdots, T. \tag{11}$$

Since the servers are all off the beginning, we have

$$z_{j0} = 0,\ j = 1, \cdots, J. \tag{12}$$

Therefore, the objective is to minimize (7) which is subject to (3), (4), (8), (9), (10) and (11). And the initial conditions are given by (12).

B. Aggregating Demand to Reduce Computational Time

The number of variables for the heterogenous case, the homogenous case, and the heterogeneous homogeneous-server-cluster case, presented above are $3 \times I \times T$, $3 \times T$, and $3 \times J \times T$, respectively. Since the computational time grows with the number of integer variables, the time complexity for the heterogenous and heterogeneous homogeneous-server-cluster cases differs significantly for larger scale data centers.

In this dissertation, we propose aggregating a certain number of continuous time slots into a single aggregated slot to reduce the total number of time slots for the workload. The value of workload demand on an aggregated time slot is decided by the workload of the original time slots and the service-level agreement (SLA). Let $\hat{T}$ denote the reduced number of time slots. We propose two aggregation modes and two aggregation methods (static and dynamic) for each mode.

1) Aggregation by Maximum: If an SLA stringently requires that the workload should be satisfied all the time, then the workload of an aggregated time slot takes the maximum of the demand of the original time slots. For example, we want to aggregate $[d_k, d_{k+1}, \cdots, d_{k+\ell}]$ into one time slot; then the new demand $\hat{d}_k$ with ℓ times of the original slot size is given by

$$\hat{d}_k = \max\{d_k, d_{k+1}, \cdots, d_{k+\ell}\}. \tag{13}$$

This method introduces an artificial increase in the demand, which is not needed by users. Increasing demand causes extra consumption of power energy. On the other hand, aggregation smoothes out the "regularity" of the workload that has effects on the switching cost. That is, we trade off the time complexity of running the model at the expense of extra costs on energy consumption.

We can aggregate every M contiguous time slots into one. This is called static aggregation. We have $\hat{T} = \lceil T/M \rceil$. The slot size of the aggregated workload (except the last slot) is M times of the slot size of the original demand. The slot size of the last slot is $T - M \times (\hat{T} - 1)$. The aggregated workload $\lceil t/M \rceil$ is given by (except the workload for the last slot which is trivial)

$$\hat{d}_{\lceil t/M \rceil} = \max\{d_{(\lceil t/M \rceil - 1)\cdot M + 1}, \cdots, d_{(\lceil t/M \rceil - 1)\cdot M + M}\} \tag{14}$$

Although some waste of energy consumption is inevitable, we can alleviate waste by using dynamic aggregation. Instead of aggregating the workload in equal numbers of original time slots statically, the adaptive aggregation method aggregates an arbitrary number of time slots as long as the number of aggregated time slots is $\hat{T}$ such that the sum of the difference between the aggregated workload and the original workload is minimized. That is, we need to minimize

$$\sum_{k=1}^{\hat{T}} \sum_{\ell=1}^{T_k} (d_k^{max} - d_{kl}),\ \text{where } d_k^{\max} = \max\{d_{k1}, \cdots, d_{kT_k}\},\ (15)$$

that is subject to

$$\sum_{k=1}^{\hat{T}} T_k = T. \tag{16}$$

To this end, we need to choose $\hat{T} - 1$ review points out of $T - 1$, requiring $\binom{T-1}{\hat{T}-1}$ operations. This extra computational time complexity is contradictory to the purpose of doing aggregation. Thus, we propose *local smooth* heuristics to implement this idea. We call this procedure *improved local smooth algorithm* (see Algorithm 1). In Algorithm 1, $\mathcal{O}(T)$ complexity is achieved.

Algorithm 1 Improved Local Smooth Algorithm

```
1:  j ← T                                                  Initialization
2:  for i := 2 → j do
3:      si[i − 1] ← (d[i] − d[i − 1])                      Compute smooth index
4:  end for
5:  while j > T̂ do
6:      ap ← InMin(|si|)                                   Find aggregation point
7:      d[ap] ← Max/Mean(d[ap], d[ap + 1])
8:      d[ap + 1] ← Max/Mean(d[ap], d[ap + 1])             Aggregate
9:      si[ap − 1] ← d[ap] − d[ap − 1]                     Update smooth index
10:     si[ap] ← d[ap + 2] − d[ap + 1]                     Update smooth index
11:     ss[ap] ← ss[ap] + ss[ap + 1]                       Compute the size of
        aggregated slots
12:     if ap ≠ j − 1 then
13:         for k := ap + 1 → j − 1 do
14:             d[k] ← d[k + 1]  Adjust the index of slots behind the
                aggregation point
15:             ss[k] ← ss[k + 1]  Adjust the corresponding slot size
16:             if k < j − 1 then
17:                 si[k] ← si[k + 1]                      Adjust the smooth index
18:             end if
19:         end for
20:     end if
21:     j ← j − 1                                          Decrease the number of slots by 1
22: end while
23: return d, ss
```

However, since it is based on the local information, there is no guarantee that this method will reach the global optimal solution. In some rare cases, it is possible that this solution is worse than static allocation in terms of the optimum. Note that the computational time for the dynamic aggregation has no significant difference with its static aggregation counterpart since they have the same number of variables and constraints. To differentiate the proposed dynamic aggregation and implemented dynamic aggregation, we call the proposed dynamic aggregation as *strict dynamic aggregation.*

2) Aggregation by Mean: The workload demand of the aggregated time slot can also take a certain percentile of workload demand of the original slots. "Aggregation by maximum" actually takes 100-th percentile of the original slots. For many long-live jobs, which do not need to be executed on real-time, such as data warehousing and scientific computation, the workload can be arranged over time as long as the average workload over time is satisfied. In this paper, we introduce aggregation by mean: the workload demand of the aggregated time slot takes the mean of the workload demand of the original slots. For example, We

aggregate $[d_k, d_{k+1}, \cdots, d_{k+\ell}]$ into one time slot, then the new demand $\hat{d}_k$ with ℓ times of the original slot size is given by

$$\hat{d}_k = \text{mean}\{d_k, d_{k+1}, \cdots, d_{k+\ell}\}. \quad (17)$$

Compared with aggregation by maximum, aggregation by mean does not introduce the artificial increase of the workload demand and smoothes out the regularity of the workload.

Similar to the static aggregation in aggregation by maximum, we aggregate every M continuous time slots into 1. The only difference is that the aggregated workload takes the average of original workload:

$$\bar{d}_{\lceil t/M \rceil} = \text{mean}\{d_{(\lceil t/M \rceil - 1)\cdot M+1}, \cdots, d_{(\lceil t/M \rceil - 1)\cdot M+M}\} \quad (18)$$

The application of this method is based on the ability of re-arranging the user request within a certain time limitation for some applications that do not requires real-time execution. In other words, the requested load can be either executed in advance or delayed in the data center.

To improve the user experience, we need to avoid delay or advance workload as much as possible. Therefore, we have the counterpart of dynamic aggregation of what we have in aggregation by maximum. The objective function is to minimize

$$\sum_{k=1}^{\hat{T}} \sum_{\ell=1}^{T_k} |d_{kl} - \bar{d}_k|, \text{where} \bar{d}_k = \text{mean}\{d_{k1}, \cdots, d_{kT_k}\} \quad (19)$$

which is subject to (16).

It is worthy to note that although (15) and (19) look similar, the objectives of dynamic aggregation in aggregation by maximum and aggregation by mean are different. In aggregation by maximum, the aim is to reduce "wasted energy". In aggregation by mean, the target is to reduce the movement of workload. Thus, compared with static aggregation, in aggregation by maximum, it results in less energy costs while in aggregation by average, the energy cost is always the same.

However, the proposed dynamic aggregation method has no constraints on the number of original slots for the aggregated slot. In practical applications, the workload can only be executed in advance or delayed for a certain time. Let S be the maximum number of continuous time slots that can be aggregated. Consequently, this problem is also subject to

$$\max\{s_1, \cdots, s_{\hat{T}}\} \leq S. \quad (20)$$

The exact solution based on Improved Local Smooth algorithm requires $n!$ time. We suggest a low complex approximation that relaxes the target number of aggregated workload slots to guarantee the movement of the workload is less than a certain threshold. The modified algorithm is described in the dissertation. The problem with this implementation is that it may not have a solution. We can swap line 22 and 23 in the algorithm to relax the target number of time slots to guarantee there is a solution.

C. Results

Due to the space limit, we only summarize the key points of the results in this paper and refer readers to the dissertation [34] for the details of experiments and results. Our numerical results show that aggregation is an efficient method to reduce computational time. Choosing an appropriate degree of aggregation is a tradeoff between the cost and the computational time. We observe that the dynamic aggregating method in both modes can achieve a significant gain compared with the static aggregation approach in terms of their individual objective functions. The study on varying the cost component weights shows that an appropriate degree of aggregation also depends on the weights.

constants

c_{ij}^p energy consumption of server i running at frequency option j in a time slot

c_i^{s+} cost of switching server i on

c_i^{s-} cost of switching server j off

v_{ij} capacity of server i running at frequency j

d_t demand at t

$y_{i0} = 0$ initial (time slot 0) state of server i

variables

$y_{ijt} = 1$ if server i is running at frequency option at time slot t; 0, otherwise $y_{it}^+ = 1$ if server i is turned on at time slot t; 0, otherwise

$y_{it}^- = 1$ if server i is turned off at time slot t; 0, otherwise

objective

minimize $\sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} c_{ij}^p \cdot y_{ijt} + \sum_{t\in\mathcal{T}} \sum_{i\in\mathcal{I}} \left(c_i^{s+} y_{it}^+ + c_i^{s-} y_{it}^-\right)$

constraints

$\sum_{j\in\mathcal{J}} y_{ijt} \leq 1,\ i = 1, \cdots, I;\ t = 1, cdots, T$

$\sum_{i\in\mathcal{I}} \sum_{j\in\mathcal{J}} v_{ij} \cdot y_{ijt} \geq d_t,\ t = 1, \cdots, T$

$\sum_{j\in\mathcal{J}} y_{ijt} - \sum_{j\in\mathcal{J}} y_{ij(t-1)} - y_{it}^+ + y_{it}^- = 0,\ i = 1, \cdots, I;\ t = 1, \cdots, T$

$y_{it}^+ + y_{it}^- \leq 1,\ i = 1, \cdots, I;\ t = 1, \cdots, T$

$y_{i1}^+ = \sum_{j\in J} y_{ij1},\ i = 1, \cdots, I$

$y_{i1}^- = 0,\ i = 1, \cdots, I$

Fig. 1. Binary IP: resource allocation with DVFS

III. Data Center Resource Allocation with DVFS

In this section, we add DVFS into the problem. Since each server may choose operating frequency by itself, model-Het described in Sec. II is the nature fit for this case. In addition, the static aggregation by maximum discussed in Sec. II is applied.

A. Problem Formulation

We only introduce the notations different from those used in Sec. II. 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}^p per time unit. The wear-and-tear cost of turning server i on and off is denoted by c_i^{s+}, c_i^{s-}, respectively. Let the binary decision variables y_{ijt} denote if server i is running at frequency option j at time slot t.

Due to space limits, the problem is summarized in Fig. III-A. The derivation of the formulation can be found in the dissertation [34].

B. Results

We refer readers to the dissertation [34] for details of experiments and results. Key results are summarized here. 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 degree of aggregation has pronounced effects on the optimal solutions. It is indispensable to put the problem in the multi-time period

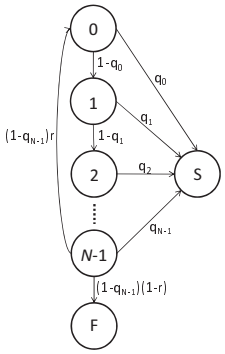

Fig. 2. Redirection strateg graph

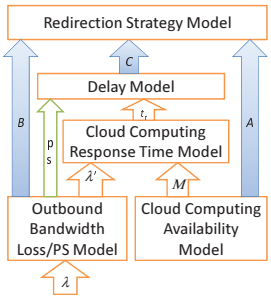

Fig. 3. Hierarchical model

framework. Aggregation is a more keen need in the case with DVFS since the computational complexity of the case with DVFS is much higher compared with the case without DVFS given other settings are the same.

IV. A Hierarchical Model to Evaluate the QoE of Online Services Hosted by Data Centers

As Internet services utilize data centers to host their service via a cloud computing model, they are challenged by evaluating the quality of experience and designing redirection strategies in this complicated environment. We propose a hierarchical modeling approach that can easily combine all components of this environment. Identifying interactions among the components is the key to construct such models.

A. The Hierarchical Model

We assume that an ISeP wants to migrate its service into cloud computing. The ISeP asks the CSP to replicate their services across multiple geographically dispersed data centers. Which user should request service from which data center is decided by the redirection strategies of the ISeP. User requests can be directed to any one of these data centers as needed using parallel serving techniques e.g., location based DNS redirection. The ISeP also wants to use more VMs than necessary to prevent the single-point-of-failure, that is, to improve availability. The hierarchical model is built from the bottom up. As shown in Fig. 3, there are two models side to side at the bottom layer. The left side is the outbound bandwidth loss model (*sub-model 1*). The probability of the outbound bandwidth being used up is denoted by B. The right side is the cloud computing availability model (*sub-model 2*). The availability of cloud computing is denoted by A. The cloud computing response time model is at the middle layer (*sub-model 3*). The CDF of the cloud computing response time is obtained by sub-model 3. According to the arrival requests with rate λ' filtered by *sub-model 1* and the number of available serving nodes (M) which are subject to *sub-model 2*, *sub-model 3* calculates the probability (denoted by C) that the response time of the cloud computing exceeds the threshold incorporated with the latency model (*sub-model 4*). At the top layer of this model, the redirection strategy graph (*sub-model 5*), which is constructed according to the redirection strategies used, sums up all the possibilities that the request being denied (A, B, and C) from the sub-models under it. Due to the space limit, the derivation of each sub-model is omitted.

The ISeP can implement any redirection functionality. Therefore, the ISeP can utilize redirection strategies to increase the success probability. In the dissertation, we discuss and compare strategies. The promised QoE decides how to apply these strategies. For instance, the product of the maximum number of tries and the time threshold (without considering retry probability) should be bounded by the user tolerable latency. We use the redirection strategy graph to model these strategies. As shown in Fig. 2, an ISeP sends a user request to the data center (replicated service) 0 with probability q_0 of success and probability $1 - q_0$ of failure. If the user is not successfully served, the ISeP then redirects the request to data center 1 with probability q_1 of success and so forth, until reaching the maximum number of data centers. Then, the request can be redirected to the data centers that have been visited with the probability of $r \cdot (1 - q_j)$ for $j = 0, 1, \cdots, N$, where $(1 - q_j)$ can be found in the dissertation [34]. This redirection strategy graph is a discrete time Markov chain (DTMC) with two absorbing states, S and F. Solving two metrics: average time to complete and complete probability according to the redirection strategy graph is derived in the dissertation [34].

B. Results

In the dissertation, we present a set of numeric results derived from our model to understand the different implications in this environment. This model is implemented in the SHARPE package [42]. We find there is an optimal number of VMs in terms of availability. We illustrate the settings for cloud computing to achieve optimal QoE. We also quantify the difference between choosing data centers from the nearest to the furthest and choosing data centers randomly. The modeling approach can help data center operators allocate resources and design redirection strategies to services or help service providers to evaluate infrastructure providers.

V. Conclusion

This dissertation has solved two data center resource management problems. The first one is to answer how to minimize server operational costs by leveraging switching servers on/off and DVFS when resource requirements are given. The dynamics of the workload demand must be aware of in this particular problem to avoid local optimum. We first consider switching servers on/off. Our evaluation shows that our approaches can save significant operational costs. To alleviate the computational complexity, we propose aggregating workload demand. It shows that the increase of optimum and the decrease of computational time are at different speeds with regard to the degree of aggregation. The dynamic aggregating method can achieve significant gain (energy saving for aggregation by maximum and workload rearrangement for aggregation by mean) compared with the static aggregation approach. Adding DVFS significantly reduces data center operational costs and increases the computational complexity.

The other one is to answer how many resources data center operators should allocate to ISePs such that user QoE is guaranteed. This dissertation has built a hierarchical model to evaluate QoE considering data center reliability, allocated outbound bandwidth, allocated buffer size, data center redirection strategies, request arrival and service pattern, *etc.* The nature of the hierarchy is suitable for rich interactions in this environment. The results of modeling can help OSPs efficiently quantify the gain of acquiring a higher performance from CSP (*i.e.*, higher reliability, more outbound bandwidth) and implementing more complicated redirection strategies while maintaining QoE.

REFERENCES

[1] "Amazon ec2," http://aws.amazon.com/ec2/. http://aws.amazon.com/ec2/

[2] M. Arlitt and T. Jin, "Workload characterization of the 1998 world cup web site," IEEE Network, Tech. Rep., 1999.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, and M. Zaharia, "Above the clouds: A berkeley view of cloud computing," EECS Department, University of California, Berkeley, Tech. Rep., 2009.

[4] L. A. Barroso, "The price of performance," *Queue*, vol. 3, no. 7, pp. 48–53, Sept. 2005.

[5] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33–37, 2007.

[6] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya, "A taxonomy and survery of energy-efficient data centers and cloud computing systems," September 2010. http://arxiv.org/ftp/arxiv/papers/1007/1007.0066.pdf

[7] J. L. Berral, n. Goiri, Í R. Nou, F. Julià, J. Guitart, R. Gavaldà, and J. Torres, "Towards energy-aware scheduling in data centers using machine learning," in *e-Energy '10: Proceedings of the 1st International Conference on Energy-Efficient Computing and Networking*. New York, NY, USA: ACM, 2010, pp. 215–224.

[8] L. Bertini, J. C. B. Leite, and D. Mossé, "Power optimization for dynamic configuration in heterogeneous web server clusters," *J. Syst. Softw.*, vol. 83, no. 4, pp. 585–598, 2010.

[9] R. Bianchini and R. Rajamony, "Power and energy management for server systems," *IEEE Computer*, vol. 37, p. 2004, 2004.

[10] M. Bichler, T. Setzer, and B. Speitkamp, "Capacity planning for virtualized servers," in *Workshop on Information Technologies and Systems (WITS)*, Milwaukee, Wisconsin, 2006.

[11] M. Blackburn, "Five ways to reduce data center power consumption," 2008. http://doe.thegreengrid.org/files/temp/E12E5A57-B5B2-71A3-DF10D563C6676C23/White_Paper_7_-_Five_Ways_to_Save_Power.pdf

[12] P. Bodik, M. P. Armbrust, K. Canini, A. Fox, M. Jordan, and D. A. Patterson, "A case for adaptive datacenters to conserver energy and improve reliability," Technial Report UCB/EECS-2008-127, EECS Department, University of California, Berkeley, Tech. Rep., 9 2008.

[13] P. Bohrer, E. N. Elnozahy, T. Keller, M. Kistler, C. Lefurgy, C. McDowell, and R. Rajamony, *The case for power management in web servers*. Norwell, MA, USA: Kluwer Academic Publishers, 2002, pp. 261–289.

[14] T. Brey and L. Lamers, "Using virtualization to improve data center efficiency," 2009.

[15] K. Chandra and A. E. Eckberg, "Traffic characteristics of on-line services," in *2nd IEEE Symposium on Computers and Communications*, July 1997, pp. 17–21.

[16] J. S. Chase, D. C. Anderson, P. N. Thakar, and A. M. Vahdat, "Managing energy and server resources in hosting centers," in *In Proceedings of the 18th ACM Symposium on Operating System Principles (SOSP*, 2001, pp. 103–116.

[17] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energy-aware server provisioning and load dispatching for connection-intensive internet services," in *Proceedings of the USENIX NSDI'08*. Berkeley, CA, USA: USENIX Association, 2008, pp. 337–350.

[18] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, "Managing server energy and operational costs in hosting centers," *SIGMETRICS Perform. Eval. Rev.*, vol. 33, no. 1, pp. 303–314, 2005.

[19] E. M. Elnozahy, M. Kistler, and R. Rajamony, "Energy-efficient server clusters," in *In Proceedings of the 2nd Workshop on Power-Aware Computing Systems*, 2002, pp. 179–196.

[20] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *Proceedings of the 34th annual international symposium on Computer architecture*, ser. ISCA '07. New York, NY, USA: ACM, 2007, pp. 13–23.

[21] D. Filani, J. He, S. Gao, M. Rajappa, A. Kumar, R. Shah, and R. Naappan, "Dynamic data center power management: Trends, issues and solutions," *Intel Technology Journal*, 2008.

[22] D. Gmach, J. Rolia, L. Cherkasova, and A. Kemper, "Workload analysis and demand prediction of enterprise data center applications," in *Proceedings of the 2007 IEEE 10th International Symposium on Workload Characterization*, ser. IISWC '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 171–180.

[23] B. Hayes, "Cloud computing," *Communications of the ACM*, vol. 51, pp. 9–11, 2008.

[24] M. Kalyanakrishnan, R. Iyer, and J. Patel, "Reliability of internet hosts - a case study from the end user's perspective," *Computer Communications and Networks, International Conference on*, vol. 0, p. 418, 1997.

[25] R. Kohavi, "Practical guide to controlled experiments on the web: Listen to your customers not to the hippo," in *In ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2007*, 2007, pp. 959–967.

[26] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster Computing*, vol. 12, no. 1, pp. 1–15, 2009.

[27] A. Li, X. Yang, S. Kandula, and M. Zhang, "CloudCmp: Shopping for a cloud made easy," in *Proc. of 2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud)*, June 2010.

[28] T. Lu and M. Chen, "Simple and effective dynamic provisioning for power-proportional data centers," http://arxiv.org/abs/1112.0442, Dec. 2011.

[29] V. Petrucci, O. Loques, and D. Moss, "A dynamic configuration model for power-efficient virtualized server clusters," 11th Brazilian Workshop on Real-Time and Embeded Systems, May 2009. http://www.ic.uff.br/~vpetrucci/papers/petrucci09wtr.pdf

[30] V. Petrucci, O. Loques, and D. Mossé, "Dynamic configuration support for power-aware virtualized server clusters," Tech. Rep., 2009, technical Report.

[31] V. Petrucci, O. Loques, and D. Mossé, "Dynamic optimization of power and performance for virtualized server clusters," Technical Report, 2009. http://www.ic.uff.br/~vpetrucci/papers/pado-report.pdf

[32] E. Pinherio, R. Bianchini, E. V. Carrera, and T. Heath, *Compilers and Operating Systems for Low Power*. Kluwer Academic Publishers, 2003, ch. Dynamic Cluster Reconfiguration for Power and Performance.

[33] H. Qian, S. Dispensa, and D. Medhi, "Optimizing request denial and latency in an agent-based VPN architecture," in *IEEE/IFIP Network Operations and Management Symposium (NOMS) 2008*, Salvador, Brazil, April 2008, pp. 248–255.

[34] H. Qian, "Data center resource management with temporal dynamic workload," Ph.D. dissertation, University of Missouri-Kansas City. http://hdl.handle.net/10355/14696

[35] H. Qian, S. Dispensa, and D. Medhi, "Balancing request denial probability and latency in an agent-based vpn architecture," *IEEE Transactions on Network and Service Management*, vol. 7, no. 4, pp. 282–295, 2010.

[36] H. Qian, F. Li, and D. Medhi, "On energy-aware aggregation of dynamic temporal demand in cloud computing," in *COMSNETS*, 2012, pp. 1–6.

[37] H. Qian and D. Medhi, "Estimating optimal cost of allocating virtualized resources with dynamic demand," in *Proceedings of the 23rd International Teletraffic Congress*, ser. ITC '11. ITCP, 2011, pp. 320–321.

[38] H. Qian and D. Medhi, "Server operational cost optimization for cloud computing service providers over a time horizon," in *UNSENIX Hot'ICE 2011*, 2011.

[39] H. Qian, D. Medhi, and K. S. Trivedi, "A hierarchical model to evaluate quality of experience of online services hosted by cloud computing," in *Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management, IM 2011, Dublin, Ireland, 23-27 May 2011*, N. Agoulmine, C. Bartolini, T. Pfeifer, and D. O. Sullivan, Eds. IEEE, 2011, pp. 105–112.

[40] H. Qian, C. S. Surapaneni, S. Dispensa, and D. Medhi, "Service management architecture and system capacity design for phonefactor: a two-factor authentication service," in *Proceedings of the 11th IFIP/IEEE international conference on Symposium on Integrated Network Management*, ser. IM'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 73–80.

[41] H. Qian, R. S. Surapaneni, M. Ray, S. Dispensa, and D. Medhi, "Dreamcache: Distributed real-time transaction memory cache to support two-factor authentication services and its reliability," in *IEEE/IFIP Network Operations and Management Symposium 2010*, 2010.

[42] R. Sahner, K. Trivedi, and A. Puliafito, *Performance and Reliability Analysis of Computer Systems: An Example-Based Approach Using the SHARPE Software Package*. Kluwer Academic Publishers, 1995.

[43] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," in *Proceedings of the 2008 conference on Power aware computing and systems*, ser. HotPower'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 10–10.

[44] _____, "Epa report on server and data center energy efficiency," U.S. Enviromental Protetion Agency, 2007, eNERGY STAR Program.

[45] A. Verma, G. Dasgupta, T. K. Nayak, P. De, and R. Kothari, "Server workload analysis for power minimization using consolidation," in *Proceedings of the 2009 conference on USENIX Annual technical conference*, ser. USENIX'09. Berkeley, CA, USA: USENIX Association, 2009, pp. 28–28.

[46] K. V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability," in *Proc. of 1st ACM Symposium on Cloud Computing*, June 2010.

[47] E. Wustenhoff, "Serice level agreement in the data center," http://www-it.desy.de/common/documentation/cd-docs/sun/blueprints/0402/sla.pdf.

[48] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.