Hierarchical SLA-Driven Resource Management for Peak Power-Aware and Energy-Efficient Operation of a Cloud Datacenter

Hadi Goudarzi[1] and Massoud Pedram[2], *Fellow, IEEE*
[1] Qualcomm Incorporated, CA Email:*hgoudarz@usc.edu*
[2] University of Southern California, Department of Electrical Engineering, CA Email:*pedram@usc.edu*

Abstract -- **In this work, a hierarchical, service level agreement (SLA) based resource management solution for cloud datacenters is presented, which considers the energy non-proportionality of existing servers, peak power constraints, and cooling power consumption. The goal of this resource manager is to minimize the operational cost of the data center. The hierarchical structure of the proposed solution makes the resource management scalable. The proposed resource management solution simultaneously considers server and cooling power consumption, guarantee-based SLA and complexity of the decision making in the resource management of the cloud computing systems. Considering SLA and state of the datacenter in determining the amount of resource that needs to be allocated to applications results in significant reduction of the operational cost in datacenter. The effectiveness of the proposed management scheme compared to previous work is demonstrated using a comprehensive cloud computing simulation tool. The proposed resource management algorithms reduce the operational cost of a datacenter by about 40% while satisfying SLA constraints and decrease the run-time of the management algorithms by up to 86% with respect to the state of the art centralized management solution.**

I. INTRODUCTION

Demand for computing power has been increasing and datacenters are now faced with a major impediment of power consumption. Some reports such as [1] and [2] estimate the datacenter electricity demand in 2012 was around 31 GW globally which is equivalent to the electricity demand of around 23 million homes. These reports also predict fast growth rate for electrical energy consumption in datacenters. Resource over-provisioning and energy non-proportional behavior of today's servers [3] are two of the most important reasons for high energy consumption of datacenters. In addition to these factors, limited capacity of the power delivery network (PDN) in the datacenter facility and inefficiency and power dissipation of the power backup and distribution subsystem and the computer room air conditioning (CRAC) subsystem makes the problem of power management in datacenter challenging.

The IT infrastructure provided by the datacenter owners/operators must meet various SLAs established with the clients. SLAs include constraints on provided performance level or physical resources such as compute power, storage space, network bandwidth, availability and security, etc. Infrastructure providers often end up over-provisioning their resources in order to meet the clients' SLAs. Such over-provisioning may increase the cost incurred on the datacenters in terms of the electrical energy bill. Therefore optimal provisioning of the resources is imperative in order to reduce the cost incurred on the datacenter operators.

There are a number of different resource managers in the datacenter. A VM manager (VMM) performs VM assignment and migration. A power manager (PM) manages the power and performance state of servers whereas a cooling manager (CM) manages the cooling and air conditioning units. In order to achieve the minimum operational cost, coordination between these managers is necessary.

The resource management policy in the cloud system is the key to determine the operational cost, client admission policy, and quality of service. Considering a given set of clients having signed appropriate SLAs with the cloud service provider, the resource management problem in the cloud system can be described as the problem of optimizing any of the aforesaid objective functions subject to the given SLAs. The resource management decisions include assigning VMs to servers, allocating resource to each VM and migrating them between servers to address SLA violations, peak power constraints or thermal emergencies.

To manage resources in a cloud system, a central manager (commissioned by the cloud service provider) can cause reliability (single point of failure) and face scalability issues. Regarding the latter point, the number of servers and VMs in a cloud system can be in the order of tens of thousands to hundreds of thousands. This underlines scalability as one of the key conditions that any resource management solution for the cloud system should satisfy. In addition to the large scale of the problem, the number of performance counters and power and temperature measurement signals from different parts of the datacenter that should be monitored to make timely decisions (for example decision about VM migration made at the millisecond rate) is huge and aggregating and analyzing this amount of data in a centralized manager may result in low performance and large energy overhead.

In this work, we propose a hierarchical and decentralized decision making architecture for VM management in cloud datacenters. The proposed solution employs a set of decentralized decision makers (managers) who are trying to solve a complex, large-scale, constrained, optimization problem with cooperation. This cooperation involves making hierarchical decisions and exchanging requests for VM assignment/migration among different managers by coordination with hierarchical power and cooling managers.

Even though different aspects of the resource management in cloud computing systems are well studied, the proposed VM management solution is the first work that simultaneously considers server and cooling power consumption, guarantee-based SLA and complexity of the decision making in VMM of the cloud computing systems. Due to the hierarchical structure of the proposed solution, the complexity of the solution is very low considering multiple-criteria decision making to increase the energy-efficiency in delivering the promised SLA. Moreover, the proposed solution considers the energy-proportionality of the servers, cooling power consumption and peak power limitation in making the resource assignment decisions. As shown in our previous work [4], considering SLA and state of the datacenter in determining VM assignment and resource allocation results in significant reduction of the operational cost of the datacenter (around 18%) compared to solving the resource assignment solution for fixed size VMs based on SLA requirements.

To show the effectiveness of the proposed management scheme, a cloud system simulation software tool has been developed. The simulator can model and do performance evaluation of both centralized and decentralized resource management architectures. Simulation results demonstrate that the decentralized resource management algorithm reduces the operational cost of a datacenter by about 40% and decreases the run-time of the algorithms up to 7 times with respect to a centralized management structure proposed in previous work.

This paper is organized as follows. The relevant prior work is reviewed in section II. The cloud system configuration and cost/performance metrics are presented in section III. The resource management problem is described in section IV. The periodic optimization strategy, a local search strategy to improve the objective function, and algorithms to handle emergency cases are presented in sections V, VI and VII respectively. Simulation framework and results are presented in section VIII whereas the paper is concluded at the last section.

II. Related Work

Resource management in datacenters and cloud systems has attracted a lot of attention in recent years. In best of our knowledge, this paper is the first research work that aims at minimizing the total energy cost of the datacenter that includes the server, IT, and, cooling infrastructure energy cost while considering the key parameters that are important in VMM including SLA and performance requirement of VMs, scalability of the solution, and peak power limitations in datacenters. Different works in the literature have attempted to solve a part of this complicated problem or propose methods to minimize the power consumption or energy cost by focusing on one or two of the mentioned aspects in this problem. In this section, a review of the most relevant work in the literature focusing on different parts of this problem is presented.

Power and performance modeling: There are different works in the literature that focus on modeling performance or power consumption of servers and in general VM consolidation in datacenter in order to be used in resource management solution. Power consumption of servers with different set of VMs assigned to them and migration latency based on the size of the VM is experimentally analyzed in [5]. Effect of uneven resource utilization and a way to improve energy efficiency by increasing the resource utilization on different resource dimensions are studied at [6]. An approach to resolve the interference between VMs placed on the same physical machine is presented at [7]. Reference [8] and [9] are examples of theoretical performance modeling in the literature.

SLA: Having clients with SLA contracts add the challenge of VM sizing to the resource management problem. Many researchers in different fields have addressed the problem of SLA-driven resource assignment. Some of the previous works consider probabilistic SLA constraints with violation penalty, e.g. [10, 4]. Some other work consider utility function based SLA [11, 12] to determine the resource allocation solution that minimizes the operational cost of the datacenter. For dynamic resource allocation solutions based on SLA, prediction of the workload is the most important step [13]. For example, reference [14] presents a resource management solution that minimizes SLA violation and energy consumption based on workload prediction in different time granularities.

Cooling power consumption: Some of the previous work (e.g. [15, 16]) consider minimizing cooling power consumption by distributing the workload in order to keep the supply cold air temperature as high as possible and avoid super-linear increase in cooling power consumption. The idea of minimizing heat recirculation using temperature-aware task scheduling (application placement) is proposed in [17]. The task scheduling policy in this work focuses on making the inlet temperature as even as possible to decrease the cooling system power consumption. Cooling-aware task scheduling in geographically distributed datacenters is presented in [18]. The authors considered batch job scheduling problem to minimize the IT plus cooling power consumption minus a fairness metric which is related to the queueing time of the tasks.

Peak power: A number of dynamic power provisioning policies have been proposed in the literature to avoid peak power emergencies, including [19, 20, 21, 22]. Fan et al. [20] present the aggregate power usage characteristics of a large datacenter over a long period. The results show a big difference between theoretical and practical peak power consumption of server clusters. This difference grows by increasing the size of the cluster. Based on the provided measurements and results, the authors outline a dynamic power provisioning policy in datacenters to increase the possibility of over-subscription of available power and protect the power distribution hierarchy against overdraw. Exploring the best way of distributing a total power budget among different servers in a server farm in order to reach the highest performance level is studied in reference [23]. Moreover, an approach to reduce the peak power consumption of servers by dynamic power allocation using workload and performance feedbacks is presented in reference [24].

Scalable VMM: Some of the previous work (e.g. [25, 26, 27]), aim to make the VMM decisions more scalable. A hierarchical resource allocation solution to minimize the server energy consumption and maximize SLA utility function is presented in [25]. The proposed hierarchical solution breaks the big problem of resource scheduling in one day to multiple smaller problems (smaller server and application set) to reduce the complexity of the problem and increase the parallelism. A decentralized VM assignment and migration is presented in [27] that targets to make the resource management solution scalable. The decision regarding accepting new VMs is decided by servers (based on a probabilistic approach) inside datacenter based on their current utilization. Even though the VM acceptance process is distributed, the process of asking from servers is centralized which can cause the performance bottleneck. Moreover, many parameters that affect the resource assignment solution such as peak power limitation and cooling system power consumption cannot be considered if only server-level resource managers are involved in make the resource management decisions.

Average power reduction: Many of the previous works are focused on minimizing the average power consumption in datacenter. Many of the work in this area are focused on getting a near energy proportional behavior from datacenter and equipment that are not designed to deliver that behavior [28]. A good example of considering server power consumption and migration cost in the resource assignment problem is reference [5]. The authors propose to use a power efficiency metric to rank the servers to find the optimal VM placement, because creating a model for all mixes of all applications on all servers is infeasible. A migration-aware heuristic based on first-fit decreasing algorithm is presented to place the applications on servers based on the introduced ranking metric.

Power manager structure: There is a strong relation between VMM and PM in datacenters. These two managers act based on the feedback from each other and both try to minimize the operational cost of the datacenter. A good example of power management structure in datacenter is presented in [19]. In that work, a hierarchical structure for power provisioning and optimization is proposed. These managers minimize the average power consumption, keep the average power consumption below temperature related power capacity and keep the peak power consumption of each component below its allowable peak power consumption determined by PDN. In some of the previous work [29, 30, 31] control theory has been applied to satisfy performance (SLA) or peak power constraints in datacenters.

III. Cloud Datacenter and Key Parameters

In the following paragraphs, the assumed architecture of the cloud datacenter is described. Next some key observations and assertions about where the system's performance bottlenecks are provided. Finally, we explain how to account for the operational cost associated with a client's VM running in the system. To increase readability, Table I presents key symbols used in this chapter along with their definitions.

A) *Cloud datacenter*

In this paper, container-based (as opposed to the older raised-floor) cloud datacenters [32] are assumed. This type of datacenter relies on containment, close-coupled cooling, and modularity to improve the datacenter's energy efficiency. An example of container-based datacenter structure is shown in Figure 1.

Containers act as separate rooms for servers. Each container includes a number of racks. Inside each rack, a number of chassis exist and each chassis comprises of a number of blade servers. d denotes the cloud datacenter. Each container, rack, chassis and blade server is identified by unique id throughout cloud datacenter, denoted by c, r, q and s respectively. The set of containers inside datacenter, set of racks inside container c, set of chassis inside rack r, and set of servers inside chassis q are denoted by C_d, R_c, Q_r and S_q respectively. We use notation $|*|$ to denote the cardinality of each set.

We assume that containers may be different from each other in terms of their rack configurations or type of blade servers deployed. However, each container is internally homogenous i.e., it employs the same blade server throughout. Servers deployed in the datacenter are chosen from a set of known and well-characterized server types. In particular, servers of a given type are modeled by their processing capacity or CPU cycles per second (C_s^p) and memory bandwidth (C_s^m) as well as their average power consumption as a function of their utilization factor. We assume that local (or networked) secondary storage (disc) is not a system

TABLE I. NOTATION AND DEFINITIONS

Symbol	***Definition***
C_d, R_c, Q_r, S_q	Set of containers, racks, chassis and servers insider datacenter, container, rack and chassis, respectively
C_s^p, C_s^m	Total processing and memory capacities of server s
P_s^0, P_s^p	Fixed and dynamic (as a function of the utilization ratio) power consumptions of server s
P_q^0, P_r^0	Fixed power consumptions of chassis q and rack r
τ	Duration of the epoch in seconds
Ψ	Electrical energy price
P_*^{PDN}	Peak power capacity of PDN at a location in the datacenter
P_d^{max}	Peak power limitation of datacenter d
PAR	Peak to average power consumption ratio
T_q^{out}, T_q^{in}	Outlet and inlet temperature of chassis q
T_{crit}	Critical temperature in datacenter
COP	Coefficient-of-performance
m_i	Required amount of memory bandwidth for VM i
mc_i^*	Migration cost of VM i in different levels ($*$ can be chassis, rack, container or datacenter)
R_i^t,f_i	Contract target response time and penalty values for each request in the SLA contract
h_i	Hard constraint on the possible percentage of violation of the response time constraint in the SLA contract
λ_i	Predicted average request rate of VM i
μ_{is}	average service rate of the VM i on server s
ϕ_{is}^p, ϕ_{is}^m	Portion of processing resources or memory bandwidth of server s that is allocated to VM i
x_s, x_q, x_r	A Boolean variable to determine if a server, chassis, or rack is ON (1) or OF (0)
y_{i*}	A Boolean variable to determine if VM is assigned to a server, chassis, or rack (1) or not (0)
z_i^*	A Boolean variable to determine if VM i is migrated in chassis level, rack level or container level (1) or not (0)
T_c^s	Supply cold air temperature in container c

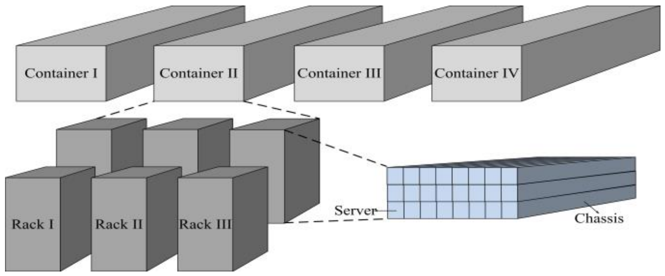


Figure 1 – An example of structure of container-based datacenter

bottleneck (although our model can easily be modified to consider this resource). The power consumption of a server is modeled as a fixed power consumption term (P_s^0) plus another variable power consumption term, which is linearly related to the utilization of the server (with slope of P_s^p). Similar to servers, each chassis and each rack consume a fixed power if they are active. These power consumptions are denoted by P_q^0 (accounting for the fans and power regulators inside each chassis) and P_r^0 (accounting for fan, power regulators and networking gears inside rack).

A network of high-efficiency uninterruptable power supply (UPS) units is used to connect the power supplies (from utility companies or generated in datacenter's site) to the datacenter's PDN. Electric power is fed to the datacenter using a PDN, which is structured in a hierarchy similar to the one used to organize servers in the datacenter. As a result, each chassis, each rack, and each container have specified peak power capacities (cap). Ψ denotes the *electrical energy price* for the datacenter.

There are two different sources for power unavailability at some location in the datacenter: 1) PDN bottleneck and 2) limitation on the total provided power to the datacenter. The first problem is much more serious than the second one [20]. The peak power capacity of PDN at a location in the datacenter is shown by P_*^{PDN} where $*$ denotes the whole datacenter, some container, some rack, or some chassis. The peak power limitation imposed by the UPS inadequacy, local electricity generation constraint, or limitation on the provided power by utility companies is shown by P_d^{max}. Determining the peak power consumption for a mix of applications in a server, chassis, rack, container or the whole datacenter is even more arduous. There are different studies focused on this issue to determine the power provisioning policy in a datacenter e.g. [20] and [22]. In this work, we assume that the peak power consumption at each granularity level of a datacenter can be estimated by multiplying the average power consumption and a factor related to the mix of running applications at that level. This factor can be large in case of homogenous workload mixes but it decreases if the heterogeneity of the workload in the mix goes up [20]. This factor can be calculated based on profiling and/or prediction methods, as suggested in [20]. We call this factor the *peak to average ratio*, (PAR). It is calculated for each level of granularity, ranging from chassis, rack, and container to datacenter.

Cooling in each container is accomplished in at least one of three of ways: overhead Cooling, in-row cooling, and circular in-row cooling [33]. A big portion of the total power consumed in a datacenter (up to 30% in older datacenters [34]) is related to cooling infrastructure. The cooling power consumption is non-linearly proportional to the total power usage in the container/datacenter. We assume that the air flows in different containers are isolated from each other and each container has its own CRAC unit. The temperature spatial granularity considered in this work is at the chassis level. Cold air is drawn to each chassis with temperature T_q^{in} and exits from the other side with temperature T_q^{out}. T_q^{in} is a function of the supply cold air temperature (T_c^s) of the container and recirculation of the heat from other chassis in that container. Similar to the work by Tang et al. [17], the recirculation of heat can be described by a cross-interference matrix, which is shown by $\boldsymbol{\phi} = \left[\phi_{ij}\right]_{N_c \times N_c}$ in which N_c is the number of chassis in container c. According to reference [17], the vector of input cold air temperatures ($\vec{T}^{in}$) in a container can be calculated based on the following formula.

$$\vec{T}^{in} = T_c^s + \boldsymbol{D}\vec{P}_c, \boldsymbol{D} = [(\boldsymbol{K} - \boldsymbol{\phi}^T\boldsymbol{K})^{-1} - \boldsymbol{K}^{-1}] \qquad (1)$$

where $\vec{P}_c$ vector denotes power consumption of chassis in container c, and $\boldsymbol{K}$ is an $N_c \times N_c$ diagonal matrix whose entries are thermodynamic constants of different chassis. The cooling manager in datacenter makes sure that T_q^{in} for each chassis is less than a pre-specified critical temperature (T_{crit}). Note that, supply cold air temperature is determined based on this constraint. *Coefficient of performance* (COP), which determines the efficiency of the cooling system, is a monotonically decreasing function of of T_c^s. Due to high efficiency of transmission and conversion efficiency in PDN of today's datacenters, we consider $1 + 1/COP(T_c^s)$ to represent the power usage effectiveness for each container. The proposed algorithm assumes a general COP function but for simulation environment we used the derived function in reference [17].

B) Virtual machine characteristics

In this work, we consider virtualized datacenters. Each client of the cloud system owns a VM that typically runs one or more applications. Each VM is identified by a unique identifier, represented by index i.

Different resources in servers such as the processing cores, memory, communication bandwidth, and secondary storage can be allocated between assigned VMs by a fixed or round-robin scheduling policy. The amount of allocated resource to a VM is a function of the VM type and the client's SLA contract. In this work, we consider the processing unit and memory bandwidth to have fixed allocation policy. ϕ_{is}^p denotes the portion of the processing capacity of server s that is allocated to VM i. The amount of memory allocated to a VM does not significantly affect performance of the VM under different workloads as long as it is no less than a specified value [35]. Hence, we assign a fixed amount of memory (m_i) to the i^{th} client's VM on any server that the VM is assigned to.

Migrating a VM between two servers causes a downtime in the client's application. Duration of the downtime is related to the migration technique used in the datacenter and the communication distance between the source and destination of the move. We assume that there is a defined cost in SLA contracts for these short but infrequent service outages based on the length of the downtime. mc_i^* denotes the part of the migration cost of the VM i related to the level of the migration where $*$ can be chassis, rack, container or datacenter. For example, the cost of migrating VM i from one rack to another rack inside one container can be calculated as $mc_i^Q + mc_i^R$.

In this work we focus on response time-sensitive applications. Clients in a cloud system have specific SLAs with the cloud provider. SLA sets a target performance for the client's instances of the application runs and requires that the cloud service provider meet that target response time (R_i^t) for no less than a certain percentage of the runs ($1 - h_i$), e.g.

95%. Furthermore, the service provider has to pay a penalty (f_i) for any application run that violates its performance target.

To estimate response time of an application, a performance model must be considered. To model the response time, we assume that the inter-arrival times of the requests for each application follow an exponential distribution function similar to the inter-arrival times of the requests in the e-commerce applications [10]. To calculate the resource requirements, the average inter-arrival rate (λ_i) of the requests for each client in each time window is predicted for the optimization procedures.

A *multiclass single-server* (MCSS) *queue* exists in servers that provide service to more than one VM. We consider *generalized processor sharing* (GPS) model at each queue; The GPS model approximates the scheduling policy used by most operating systems, e.g., weighted fair queuing and the CPU time sharing of Linux. Using this scheduling policy, MCSS queue can be replaced by multiple single-server queues. Note that the processing capacity of server s allocated to VM i is calculated as $C_s^p \phi_{is}^p$. Furthermore, it is assumed that client service times follow an exponential distribution. Let μ_{is} denote the average service rate of the VM i on server s when it has a unit of the server's processing capacity. Now then, the response time (R_i) of a VM follows an exponential distribution function with a mean value, which is calculated as follows:

$$\bar{R}_{is} = \frac{1}{C_s^p \phi_{is} \mu_{is} - \lambda_i} \quad (2)$$

We assume that μ_{is} for all servers of the same type are equal. To determine μ_{is}, we consider an offline profiling mechanism which collects service rates for different types of VMs running on different types of servers. These values indeed capture the compatibility of certain client types to certain server types.

Considering these models for request arrival and service rate, SLA-related response time constraint for each VM (i.e. $prob\{R_i > R_i^t\} < h_i$) can be expressed as follows:

$$e^{-(C_s^p \phi_{is}^p \mu_{is} - \lambda_i) R_i^t} \le h_i \Rightarrow \phi_{is}^p \ge (\lambda_i - \ln h_i / R_i^t) / \mu_{is} C_s^p \quad (3)$$

In addition to parameters for the cloud system and clients, there is only one decision parameter: ϕ_{is}^p. This parameter determines which server is chosen for VM i and how much of the processing resource is allocated to that VM. In order to simplify the resource management problem formulation and make it more understandable, we use a few more parameters as follows. ϕ_{is}^m denotes the amount of memory bandwidth allocated to the VM in server s; x_s is a Boolean variable that determines whether the server is on or off; y_{is} signifies whether VM i is assigned to server s or not; and z_i^Q determines if VM i is migrated in chassis level or not. As shown in the next section, these parameters can be derived from ϕ_{is}^p. Chassis, rack, container, and datacenter level version of x_s, y_{is}, and z_i^Q are also used in the problem formulation. In VMM problem, T_c^s is also a derived parameter that satisfies the temperature constraint of the container based on the VM assignment solution.

IV. Cloud Datacenter VM Management Problem

The VM management problem in cloud systems is the key to determining the operational cost and the quality of service.

Considering a set of clients with SLAs with the cloud provider, the VM management problem in a cloud system can be described as the problem of assigning VMs to servers and allocating resources to them to minimize the total operation cost of the cloud and the SLA violation penalties subject to performance and resource availability constraints. The biggest part of the operation cost of a datacenter is the electrical energy cost, which must be paid to the utility companies providing the electricity. To minimize the power consumption in each datacenter, the number of active or idle servers should be reduced and at the same time, the power consumption of the active servers should be balanced as much as possible in order to reduce the cooling system's power consumption.

Notice that from the VM management solution, an expected quality of service (QoS) provided to each client is calculated. Based on this QoS, expected SLA violation penalties for all clients can be computed. The other type of penalty that should be paid to the customers is the result of VM migration between different servers, which can result in service outage for a short period of time.

Various resource managers in cloud datacenters perform their jobs by interacting with each other. The VMM performs the resource management interacting with the PM and the CM. The PM reduces the power consumption in the system and uses control-theoretic solutions to ensure that the peak power of each component remains below the given peak power capacity. The CM reduces the cooling system power consumption subject to keeping the temperature below a critical threshold at every location inside datacenter. The focus of this work is VMM that considers constraints from PM and CM and optimizes the operational cost of the data center.

The common approach used in VMM is to perform optimization periodically (such a period is called *epoch* with duration τ) and modify the solution during each epoch in case of SLA, peak power or temperature emergencies or dramatic workload changes. These processes are called periodic and reactive optimizations, respectively. In periodic optimization, prediction of workload for each VM in addition to VMs' expected behavior upon assignment to different types of servers is used to determine the VM assignment and resource allocation solution.

To assign VMs to datacenters in a multi-datacenter cloud system, the whole application run-time, which can last for multiple decision epochs, should be considered. Therefore, the cloud manager needs to consider the workload trend for each client as well as the energy price during the day in order to decide how to assign VMs to datacenters in a multi-datacenter cloud system. These decisions are usually made based on cloud service provider's policy that aims to achieve some kind of geographical load balancing [36]. In this work we focus on the VMM problem in one cloud datacenter, considering only the resource assignment solution in the previous epoch to minimize the operational cost for the current epoch. Resource assignment parameters related to the previous epoch are marked with superscript γ. Periodic VM management problem can be formulated as follows:

$$Min\ \tau\Psi\sum_{c\in C_d}P_c+\sum_i f_i\tau\lambda_i\sum_s y_{is}e^{-(C_s^p\phi_{is}^p\mu_{is}-\lambda_i)R_i^t}+\sum_i\sum_{*\in\{Q,R,C,D\}}mc_i^*z_i^*$$

subject to:

$$x_s\geq\sum_i\phi_{is}^p,\ x_q\geq\frac{\sum_{s\in S_q}x_s}{|S_q|},x_r\geq\frac{\sum_{q\in Q_r}x_q}{|Q_r|},x_*\in\{0,1\} \tag{4}$$

$$\phi_s^p=\sum_i\phi_{is}^p,\phi_s^m=\sum_i\phi_{is}^m,0\leq\phi_s^*\leq 1 \tag{5}$$

$$\phi_{is}^p\geq y_{is}(\lambda_i-\ln h_i/R_i^t)/\mu_{is}C_s^p,\phi_{is}^m\geq y_{is}m_i/C_s^m \tag{6}$$

$$\begin{cases} y_{is}\geq\phi_{is}^p,\sum_s y_{is}=1\\ y_{iq}=\sum_{s\in S_q}y_{is},y_{ir}=\sum_{q\in Q_r}y_{iq}\\ y_{ic}=\sum_{r\in R_c}y_{ir},y_{i*}\in\{0,1\}\end{cases} \tag{7}$$

$$\begin{cases} z_i^Q=\sum_q\sum_{s\in S_q}\left(y_{is}-y_{is}^{\gamma}\right)^+,z_i^R=\sum_r\sum_{q\in Q_r}\left(y_{iq}-y_{iq}^{\gamma}\right)^+\\ z_i^C=\sum_c\sum_{r\in R_c}\left(y_{ir}-y_{ir}^{\gamma}\right)^+,z_i^D=\sum_d\sum_{c\in C_d}\left(y_{ic}-y_{ic}^{\gamma}\right)^+\end{cases} \tag{8}$$

$$\begin{cases} P_q=P_q^0x_q+\sum_{s\in S_q}x_s\left(P_s^0+P_s^p\phi_s^p\right)\leq\frac{P_q^{PDN}}{PAR_q}\\ P_r=P_r^0x_r+\sum_{q\in Q_r}P_q\leq\frac{P_r^{PDN}}{PAR_r}\\ P_c=\left(1+\frac{1}{COP(T_c^S)}\right)\sum_{r\in R_c}P_r\leq\frac{P_c^{PDN}}{PAR_c}\\ \sum_{c\in C_d}P_c\leq\min\left(\frac{P_d^{PDN}}{PAR_d},\frac{P_d^{max}}{PAR_d}\right)\end{cases} \tag{9}$$

$$T_q^{in}\leq T_{crit} \tag{10}$$

where $(A)^+$captures the maximum value between A and 0.

There are three main terms in the objective function: (i) IT and cooling energy cost, (ii) SLA violation penalty, and, (iii) SLA penalties related to service outage caused by VM migration.

Constraint (4) determines whether or not the server, chassis or rack is active. Constraint (5) determines the utilization of each server and forces them to be less than one. Constraint (6) determines the lower bound on the processing and memory bandwidth share of a VM from their host machine based on SLA constraint. Constraint (7) determines the assignment parameters (assignment of a VM to a server, chassis, rack and container). Although the assignment parameter for chassis to container can be determined directly from the assignment parameter for servers, these parameters are derived to be used in constraint (8) to capture the VM migration cost. Results of these constraints are z_i^* parameters that determine whether or not a VM is migrated in chassis level, rack level or container level. Constraint (9) calculates the average power consumption of the chassis, rack, container and datacenter and limits the corresponding power consumption to be less than the power provisioning capacity in PDN. This constraint also limits the peak power consumption of the datacenter to the maximum provided power. Constraint (10) forces the inlet temperature of each chassis to be lower than the critical temperature. T_q^{in} is a function of $\vec{P}_c$ that can be calculated from (9).

Periodic resource management problem is a mixed-integer non-linear programming problem. By some simplification, bin-packing problem and generalized assignment problem can be reduced to this problem. So, this problem is an NP-hard problem.

In this work, VMM utilizes the constraints and objectives in CM and PM in order to come up with a VM assignment and resource allocation that optimizes the true energy cost in the system and satisfies the SLA, peak-power and temperature constraints. This consideration also results in smaller number of VM migration due to resource limitations. Ignoring the peak-power constraints and cooling power consumption and temperature constraints can lead to inefficient use of resources and instability in creating a feasible solution by VMM.

A set of hierarchical and decentralized decision makers fit the distributed nature of resources in cloud systems. Hierarchical structures for power management and reactive management have been proposed in the previous work, c.f. [19] and [26]. Moreover, a centralized manager can cause reliability (single point of failure) and scalability issues. The big number of servers and VMs in large datacenters emphasizes scalability as one of the most important factors in designing resource managers. In addition to big scale of the problem, the number of performance counters and power and temperature measurement signals from different parts of a datacenter is huge and aggregating this amount of data in a centralized manager may result in low performance and large overhead. Finally, there are certain management functions (e.g., doing VM migration in case of power or temperature emergencies) that are best handled by local managers.

In this work, we present hierarchical resource management (HRM for short) solution in a cloud system. A figurative architecture for this manager is shown in Figure 2. This hierarchy includes a cloud manager, datacenter managers, container managers, rack managers and chassis managers. The hierarchical managers collectively try to solve a constrained optimization problem with cooperation. This cooperation involves exchanging requests of resource assignment, or VM migration between resource managers in different levels. In each level of the hierarchy, VMM receives feedback from power and temperature sensors and PM and CM in that level for periodic and reactive resource management decisions.

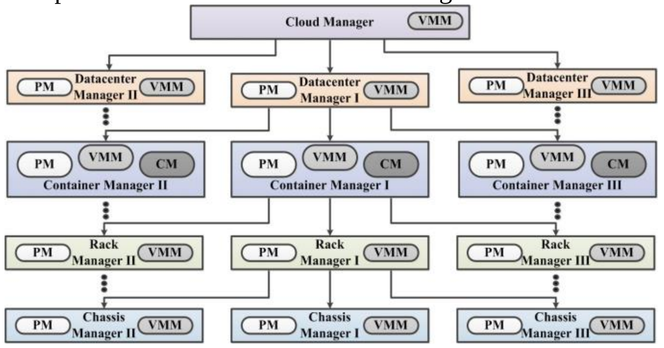


Figure 2 – An example of proposed cooperative hierarchical manager

In the proposed management architecture, periodic optimization is done by:

1- Adjusting resource allocation to active VMs
2- Assigning new VMs to servers
3- Performing local search to improve the initial solution

An abstract pseudo code for this periodic resource management is presented in Algorithm 1. To assign new VMs to servers, the status of previous VM assignment solution after proper modification is used. Instead of assigning VMs directly to servers, each resource manager distributes the VMs between its lower level resource managers and this process continues until the chassis manager assigns VMs to servers. In each resource manager, distribution of new VMs between lower-level resource managers is performed based on resource availability, peak power capacity, temperature distribution, and COP of the lower level resource managers. Due to constructiveness of this approach, a bottom-up local search procedure is used to modify the solution after assigning every VM to a server. In local search step, priority of performing a VM movement is set based on its effectiveness in reducing the operational cost of the datacenter.

Algorithm 1: Periodic Optimization Problem in VMM

```
Inputs: set of new VMs and old active VMs
Outputs: Initial VM Assignment and resource allocation for the next epoch
1  Find the best avg. workload to be used in resource manager for each VM
2  //Current VM assignment
3  Update $\phi_{is}^{p}$ for currently assigned VMs; Report $i$ to chassis if not possible
4  While (There is a VM from current set of VMs not assigned)
5    Assign reported VMs; report it to higher level manager if not possible
6  //New VM assignment
7  Determine $y_{ic}$ for all VMs in – Ranking solution in datacenter level
8  Fork ($c \in C_d$)
9    Determine $y_{ir}$ – Ranking solution in container level
10   Fork ($r \in R_c$)
11     Determine $y_{iq}$ – Ranking solution in rack level
12     Fork ($q \in Q_r$)
13       Determine $y_{is}$ – Ranking solution in chassis level
14     Join
15   Join
16 Join
17 //Local optimization
18 Fork ($c \in C_d$)
19   Fork ($r \in R_c$)
20     Fork ($q \in Q_r$)
21       Move VMs with $y_{iq} = 1$ if it decreases the objective function
22       Report a limited set of VMs with biggest move/migration incentive
23     Join
24     Move the nominated VMs if it decreases the objective function
25     Report a limited set of VMs with biggest move/migration incentive
26   Join
27   Move the nominated VMs if it decreases the objective function
28   Report a limited set of VMs with biggest move/migration incentive
29 Join
30 Move the nominated VMs if it decreases the objective function
31 Report a limited set of VMs with biggest move/migration incentive
32 End
```

Note that, the size of the problem tackled by each resource manager is much smaller than original resource assignment problem. Moreover, the assignment or local optimization in all resource managers that do not interact with each other can be executed in parallel. These two factors reduce the time complexity of the periodic optimization solution drastically. This means that this hierarchical solution makes the optimization solution more scalable without sacrificing the performance of the solution.

In case of peak power or temperature emergency, reactive optimization procedure is performed. Similar to the periodic optimization procedure, the proposed reactive optimization solution is performed in a hierarchal manner to avoid long decision making time.

VM assignment and local optimization steps in periodic optimization and reactive optimization algorithms are presented in the following sections. Some of the details, formulation and algorithms specially related to local search and reactive optimization approach are omitted due to space limitations. More details regarding HRM solution can be found on chapter 3 of reference [37].

V. Periodic optimization: VM assignmnet

The objective in the periodic optimization is to assign new VMs and re-assign active VMs to servers based on their expected workload in the next epoch so as to minimize the summation of total energy cost, the expected SLA violation penalty, and migration cost subject to resource, power and temperature constraints. The solution is strongly dependent on the existing server assignments for active VMs.

In this section, the important aspects of periodic VM assignment solution are presented:

- Abstraction of VM workload prediction to avoid under-provisioning and over-provisioning
- Finding a feasible solution for active VMs by modifying the prediction and resource allocation solution and minimal VM migration
- Abstraction of resources on each level of hierarchy for resource assignment solution for new VMs
- Important factors and resource management problem in each level of resource management hierarchy

A) *VM workload prediction*

In order to start periodic optimization, workload associated with the active and incoming VMs needs to be predicted for the next epoch. We assume that the probability distribution function of λ_i ($PDF(\lambda_i)$) for VM i in the next epoch can be predicted based on the current workload and the workload history. To account for SLA violation penalty and VM migration cost, the predicted workload ($\widehat{\lambda_i}$) can be different from the expected workload ($\overline{\lambda_i} = \int \lambda_i PDF(\lambda_i)$). In case of a large SLA violation penalty or VM migration cost, over-provisioning ($\widehat{\lambda_i} > \overline{\lambda_i}$) becomes useful. In contrast, in case of small SLA violation penalty or VM migration cost, under-provisioning ($\widehat{\lambda_i} < \overline{\lambda_i}$) may result in total cost reduction.

If $\widehat{\lambda_i}$ is used in the assignment problem, the probability of VM migration with the assumption that the selected server doesn't have any un-reserved resources can be found from the cumulative distribution function ($CDF(\widehat{\lambda_i})$). In case of migration event, cloud provider needs to pay the SLA down-time penalty in addition to the energy cost and SLA violation penalty. In order to find the best $\widehat{\lambda_i}$, we assume that the maximum tolerable λ_i based on the SLA contract (λ_i^{max}) is used to determine the resource allocation and cost of assignment in the event of VM migration to avoid another migration in the near future.

Summation of the energy cost and expected SLA violation penalty related to VM i is a function of the VM assignment and parameter λ_i used in that process. We define function $C(\lambda_i)$, which shows the minimum energy cost plus SLA violation penalty of assigning VM i to a server. To find the minimum value for the energy cost and expected SLA violation penalty for VM i, we use the maximum COP possible in datacenter and find the best ϕ_{is}^{p} for different types of servers assuming they have energy proportional behavior ($P_s^0 = 0$ and $P_s^p = P_s^p + P_s^0$). Note that in the mentioned scenario, finding ϕ_{is}^p to minimize energy cost and expected SLA violation penalty is a convex optimization problem that has a closed-form solution. The minimum energy cost and expected SLA violation penalty considering different types of servers creates $C(\lambda_i)$ function.

The predicted λ_i is the value that minimizes the expected cost in the next decision epoch:

$$\widehat{\lambda_\iota} = \underset{\lambda_i}{\operatorname{argmin}}\, CDF(\lambda_i)C(\lambda_i) + \big(1 - CDF(\lambda_i)\big)\big(\overline{mc_\iota} + C(\lambda_i^{\max})\big) \tag{11}$$

where $\overline{mc_\iota}$ is the average migration cost for the VM in datacenter.

This optimization problem tries to find the best balance between energy cost, SLA penalty cost, and expected VM migration cost. If the optimal λ_i is shown by $\widehat{\lambda_\iota}$, it can be seen that under-provisioning ($\lambda_i < \widehat{\lambda_\iota}$) results in lower operational cost but increase the probability of VM migration because $CDF(\lambda_i) < CDF(\widehat{\lambda_\iota})$. In contrast, over-provisioning ($\lambda_i > \widehat{\lambda_\iota}$) decreases the probability of VM migration but it results in higher operational cost and lower SLA violation penalty.

Consider two VMs (i and i') with the same steady characteristics and SLA contracts but different workload characteristics with the same expected average $\overline{\lambda_\iota}$. If PDF of λ_i has a longer tail compared to the one for $\lambda_{i'}$, it can be shown that $\widehat{\lambda_\iota} \geq \widehat{\lambda_{\iota'}}$. This is due to the fact that the CDF function for VM i is smaller than that for VM i' for the same λ. This in turn increases the optimum point for the function in (11) for VM i compared to VM i' to reduce the probability of VM migration.

Note that CDF and $C(\lambda_i)$ are monotonically increasing functions between 0 and λ_{max}. $\widehat{\lambda_\iota}$ can be found between possible λ_i that makes the gradient of the expected cost function in (11) equal to zero. The effect of over and under provisioning for the assignment problem is studied in the experimental results section.

B) *Finding a feasible solution for active VMs*

After finding the workload prediction for all VMs, active VMs in the previous epoch can be divided into three groups:

1) VMs that will not be active the next epoch
2) VMs with expected lighter workload in the next epoch
3) VMs with expected heavier workload in the next epoch

Allocated resources to VMs in the first category can be released. Next, resource allocation parameter (ϕ_{is}^p) for VMs with lighter workload is updated. Finally, the resource allocation parameter for VMs with heavier workload is also determined based on their current assignment although the resource requirements for some of these VMs may violate the resource constraints. Such VMs have to be migrated to different servers with more available resources.

To create an initial solution for new VM assignments problem, we use a greedy technique for these VM migrations. For this purpose, each chassis manager examines the available servers in the chassis to find a new host for the target VM. If this manager cannot find any server satisfying the VM resource requirement, it asks the parent rack manager to look for a server to host that VM and this process continues until a high-level resource manager can find a server to host the VM in question. Note that, the generated initial solution will be improved by local search after assigning every new VM to a server.

C) *Resource management problem in each level of hierarchy*

After finding an initial assignment solution for active VMs, each manager reports its current state to its (higher-level) parent resource manager. For this purpose, the chassis manager obtains the current status of all of its servers. A compacted form of this information is reported to its rack manager to model the chassis. Similarly the container manager, the datacenter manager and cloud manager can use an appropriate abstraction of the gathered information by their lower-level managers to model them in their resource management problem. In each resource manager, the lower-level entities are abstracted by one or a few servers plus the peak-power and cooling-related power consumption constraint. The assignment problem in each resource manager hierarchy can be formulated similar to the following assignment problem in datacenter-level resource manager:

$$Min\ \tau\Psi \sum_{c\in C_d} P_c + \sum\nolimits_i f_i \tau\lambda_i \sum_{c\in C_d} y_{ic} e^{-(C_s^p \phi_{ic}^p \mu_{is} - \lambda_i) R_i^t}$$

subject to resource availability and:

$$P_c = \left(1 + \tfrac{1}{COP(T_c^S)}\right)\left(P_c^\gamma + \textstyle\sum_i \phi_{ic}^p PUR_c\right) \tag{12}$$

$$\textstyle\sum_c y_{ic} = 1,\ y_{ic} \in \{0,1\} \tag{13}$$

$$\phi_{ic}^p \geq y_{ic}\left(\lambda_i - \ln h_i^c / R_i^c\right)/\mu_{is} C_s^p,\ \phi_{ic}^m \geq \tfrac{y_{ic} m_i}{C_s^m},\ \phi_{ic}^* \leq 1 \tag{14}$$

$$P_c \leq P_c^{PDN}/PAR_c \tag{15}$$

$$\textstyle\sum_i \phi_{ic}^p PUR_c\, \Delta S_c/\Delta P_c \leq S_c \tag{16}$$

$$\textstyle\sum_{c\in C_d} P_c < P_d^{max}/PAR_d \tag{17}$$

where parameter PUR_c denotes the estimated Power consumption to Utilization Ratio value for servers inside container c, parameter S_c denotes the total temperature slack (summation of $T_{crit} - T_q^{in}$ for all chassis inside the container), and, $\Delta S_c/\Delta P_c$ denotes the sensitivity of the temperature slack to the power increase in the container. Moreover, constraint (16) determines a varying temperature-related power cap on the additional power consumption in each container. Decision parameter in this problem is ϕ_{ic}^p, which determines the VM to container assignment solution (y_{ic}).

PUR parameter reflects the power consumption per unit of utilization in each server. The PUR parameter is defined to capture the energy non-proportional behavior in IT infrastructure and account for the fixed power consumptions of the server, chassis and rack. PUR parameter for server s is shown in equation (18).

$$PUR_s = P_s^p + w_s P_s^0 + w_q\, P_q^0/\left|S_q\right| + w_r\, P_r^0/|Q_r|/\left|S_q\right| \tag{18}$$

where w_* are weighting parameters that are greater than 1.

For active servers, w_* parameters are set to 1. In case of inactive server, chassis, or rack, w_s, w_q, and w_r parameters are set to the inverse of the expected utilization factor of server type, chassis, or rack. For example, if expected utilization ratio of a server type is 50%, w_s will be set to 2. This means that P_s^0 is accounted with a multiplicand of two for each unit of utilization in an inactive server with expected utilization of 50%.

Using PUR parameter gives priority to assigning VMs to active servers/chassis/rack instead of turning on a new entity. This is because active servers/chassis/racks will have lower PUR values compared to inactive ones (which must be activated to service the VM). This will result in more consolidation at the server, chassis, and rack levels. For each level of hierarchy, the weighted average PUR value of servers inside the entity is used as effective PUR value of the whole entity. Weighting is based on the remaining resource in each server dampened by the amount of remaining peak power for each entity.

Sensitivity of the temperature slack to the power increase in each chassis can be calculated using equation (1). Rack and container version of this parameter can be defined as a weighted average of the same parameter in their covered chassis. For instance, sensitivity of the temperature slack to the power increase in a container can be defined as follows:

$$\Delta S_c/\Delta P_c = \frac{\sum_{r\in R_c}\sum_{q\in Q_r}\Delta S_c/\Delta P_q\left(P_q^{PDN}-P_q\right)}{\sum_{r\in R_c}\sum_{q\in Q_r}\left(P_q^{PDN}-P_q\right)} \tag{19}$$

Finding the allocation parameter that results in minimum cost value for assigning a VM to a container is a convex optimization problem. To find this value, we need to find the derivative of the cost elements for this assignment. This derivative with respect to ϕ_{ic}^p has three elements related to the energy cost of the VM ($\tau\Psi(1+1/COP(T_c^s))PUR_c$), SLA penalty, and a term to capture the effect of this assignment on the cooling power consumption as shown in (20).

$$\frac{\partial P_c^{cooling}}{\partial \phi_{ic}^p} \cong \tau\Psi\phi_{ic}^p P_c \frac{\partial COP(T_c^s)/\partial T_c^s}{\left(1+COP(T_c^s)\right)^2}\frac{\partial T_c^s}{\partial S_c}\frac{\partial S_c}{\partial P_c}PUR_c \tag{20}$$

Note that $|\partial T_c^s/\partial S_c|$ is the inverse of the number of chassis in the container.

Solving the mentioned convex optimization problem results in finding the expected cost of assigning each VM to each container.

To solve the assignment problem based on the expected cost of assigning each VM to each container, we use a novel ranking metric. Precisely, the ranking metric for each VM, which attempts to capture the urgency of assigning the VM to the best available container for it, is calculated by subtracting the expected cost of the VM assignment to the best container (minimum cost) from the expected cost of assigning VM to the second best container. VMs are subsequently ranked based on this metric and assigned to containers with minimum assignment cost until one of the resources (CPU cycles, memory bandwidth, peak power, temperature related power cap) in one container is exhausted.

If the temperature related power cap of a container becomes zero, T_c^s is decreased and ranking parameters are updated and the container is kept as one with some extra capacity to be utilized. This process continues until all VMs are assigned to a container or resources in the datacenter are completely exhausted. In the latter case, the remaining VMs are reported to the cloud manager for re-assignment. This approach results in balanced resource utilization to reduce the power consumption with focus on *COP*, server energy efficiency and VM to server compatibility. Moreover, using PUR parameter results in giving priority in assigning VMs to containers with more active servers compared to the ones with more inactive servers. The pseudo code for this assignment solution for datacenter manager is presented in Algorithm 2.

Algorithm 2: Resource Assignment Solution

```
   Inputs: set of VMs to be assigned
   Outputs: VM Assignment to Containers
1  Foreach VM
2      Foreach $c \in C_d$
3          Calculate $\hat{\phi}_{ic}^p$ to min the cost by resource constraint
4          Calculate $Cost_{ic}$ based on $\hat{\phi}_{ic}^p$
5      End
6      $c_1 = \text{argmin}_c\, Cost_{ic}$
7      $c_2 = \text{argmin}_{c\neq c_1} Cost_{ic}$
8      $dC_i = Cost_{ic_2} - Cost_{ic_1}$
9  End
10 While (set of VMs is not empty)
11     $i = \text{argmax}_i\, dC_i$
12     Assign VM and subtract $\hat{\phi}_{ic}^p$ from available resources
13     If ($T_c^s$-related constraint limit is reached)
14         Reduce $T_c^s$ and calculate new $S_c$
15     If (peak power or server resource constraint limit is reached)
16         Remove $c$ from set of available containers
17     Update $Cost_{ic}$ and $dC_i$ for the rest of VMs if constraint limit is reached
18     Remove $i$ from set of VMs
19 End
```

A similar ranking metric and resource assignment solution is applicable in container, rack, chassis-level resource managers. Going to lower-level resource managers reduces the number of VMs to be assigned and increase the opportunity to model each entity with more than one PUR values (e.g., one for active servers and one for inactive servers) in order to increase the quality of the initial solution by giving priority to assigning VMs to active servers instead of increasing the number of active servers.

VI. Periodic Optimization: Local Search Method

Before finalizing VM assignment solution in the periodic optimization procedure, a hierarchical local search algorithm is performed to decrease the operational cost.

In the local search procedure, resource managers move VMs between servers in order to decrease SLA violation penalty or reduce the power consumption by increasing the energy proportionality or reducing the cooling power consumption. To limit the time complexity of the local search, the number of VM movement attempts is bounded. For this reason, VM movements are ranked based on their effectiveness in reducing the total cost. For each VM, a ranking metric is calculated based on the expected cost reduction from best possible movement. Depending on the resource manager, the ranking metric is found based on the most important factors on that level. SLA violation penalty, migration cost and energy proportionality are important factors in every resource manager but cooling system power consumption and temperature distribution are also important factors in the rack and container managers. The local search procedure after sorting VMs is straight-forward: (i) select the first VM; (ii) find the best real cost reduction for the VM movement; and (iii) try the VM movement if the cost reduction is positive and go to (i) if there is another VM with positive ranking metric. Note that, in case of moving a VM that was assigned to its current server in the previous epoch, VM migration cost is subtracted from cost reduction value associated with that VM. After finishing VM movements in each resource manager that can happen at the same time for all VMMs in one level of hierarchy, a limited number of VMs associated with the highest possible cost reduction values are passed to the parent resource manager and that manager start VM movements with a similar approach. Details of ranking metric for different resource managers are omitted due to space limitation.

VII. Methods to Deal with Emergencies

The periodic optimization solution cannot guarantee SLA constraint satisfaction, peak power capacity and critical temperature constraint satisfaction due to hardware failure and dynamic nature of the VM workload. A costly way of providing a performance guarantee in the cloud system is VM replication and resource over-provisioning, which are necessary for some clients (e.g., an e-commerce client) but wasteful for many others. Periodic monitoring of performance, power and temperature can be used to make reactive VM migration decisions or resource allocation adjustments in order to guarantee the satisfaction of the constraints.

Dynamic changes in VM workload is first detected by the server. Power/performance manager in each server monitors the workload of the assigned VMs and minimizes the SLA and energy cost by dynamically changing the allocation parameters. A decrease in VM workload creates power saving opportunity whereas an increase in VM workload forces the server to increase the power consumption to satisfy SLAs. Drastic increases in VM workload may not be responded by power increase and may require VM migration. Note that even if the power/performance manager module in the server can find a resource allocation solution that satisfies all of the VM SLA constraints within allowable power capacity, it is possible that migrating a VM results in lower total cost. These cases can be handled with periodic calls to local search procedure during each epoch.

There are different changes in the cloud system requiring immediate response to avoid hardware damage and huge penalties. For these situations, different control hardware components are placed inside the datacenter to avoid events such as temperature run-away in servers or power capacity violations. Some prior work such as [19] has studied the required control mechanisms. These control mechanisms can, however, result in violation of SLA constraints. The resource manager must, therefore, closely monitor the power and temperature sensors and performance counters in order to promptly migrate VM if necessary.

In different emergency scenarios (SLA, peak power or temperature emergency), in addition to the actions that can be taken by PM and CM, VM migration and resource allocation adjustment can be performed to resolve the issue. In our proposed framework, a hierarchical procedure is used to choose and then perform the best action, which results in lowest cost increase, to resolve the issue. Details are, however, omitted due to lack of space.

VIII. Simulation framework and results

A) *Simulation framework*

To show the effectiveness of the proposed resource management structure and different algorithms, a complete datacenter simulation framework is implemented in C++.

A container-based datacenter is modeled in the simulation framework. The structure of the implemented datacenters is based on the definitions in section III. Different parameters in the system are set based on real-world parameters.

We consider a heterogeneous datacenter with 16 containers and 500 servers per container. We use hourly decision epochs. Four different server types are considered in this datacenter. Processors in server types are selected from a set of Intel processors (e.g. Atom and Xeon) [38] with different number of cores, cache sizes, power consumptions and clock frequencies.

For each virtual machine in the system, a VM type from four different pre-defined VM types is selected. The virtual machine type determines the average characteristics of VM. The type also specifies variance from the mean for the matching VMs. For example, a given VM type sets μ_{is}^{p} for each server type and each VM has this service rate per unit capacity plus or minus a deviation that is solely dependent on the VM. Similarly, the average request arrival rate (which in turn determines the VM workload in each epoch) is determined by its VM type. However, the exact probability distribution function, (and thus, the variance of the distribution) depends on the specific VM. This PDF is used to determine the best workload to avoid over-provisioning or under-provisioning. Different PDF's for λ_i are used in this simulation framework including exponential and uniform distributions. In each decision epoch, the simulation starts with a randomly chosen λ_i based on the selected PDF and in each update, another λ_i is randomly chosen again based on the PDF to be used for the next time. The frequency of doing λ_i change for each VM depends on a VM-specific parameter. The number of λ_i updates for a VM can be up to 50 times per epoch. The selected λ_i in this simulation has different ranges for different VM classes and was set to be between 0.1 to 10 requests per second. To reduce the simulation time, we assume that workload changes only change the average inter-arrival time but the inter-arrival time and service time follow the exponential distribution at all time. High number of changes in average inter-arrival time makes us confident that the workload is not following the exponential distribution in general. This means that in each time, the request response time for each virtual machine can be determined based on queuing formulation. Without this assumption, the long convergence time to determine the response time for each VM does not allow us to simulate a datacenter with thousands of servers and VMs in a reasonable amount of time. The VM lifetime is set randomly based on uniform distribution between one and 8 hours.

In response to VM workload changes, the host server changes the resource allocation parameters after a small delay. For this period of time, if SLA constraint is violated, SLA violation penalty for 100% of the arriving requests is added to the cost of the datacenter. Moreover, if the server is not able to determine the resource allocation parameters such that SLA constraints for VMs or power or temperature constraint are satisfied, reactive optimizations are performed.

To simulate the datacenter, we implemented an event-driven simulator. After processing any VM workload changes, an event is generated to modify the resource allocation parameters in the server after some delay. If this resource allocation modification results in SLA violation or peak power or temperature violation, another event regarding the reactive optimizations is added to the event queue. After each event, the total cost of the datacenter is updated.

SLA for each VM is dependent on its VM type. The maximum tolerable request arrival rate, target response time, SLA violation penalty, maximum tolerable SLA violation rate, and migration penalties are determined based on the selected VM type which is set based on EC2 pricing schemes [39].

Cooling system parameters for each container are set based on the provided data in reference [17]. Peak power capacities for each component inside the datacenter (chassis, rack, and container) are pre-set as fixed parameter values. These parameters capture the ratio of the theoretical peak power consumption of each component to the actual peak power capacity. For example if this ratio is 0.8 for a chassis, it means that the peak power in that chassis cannot be more than 80% of the power consumption of an active chassis with fully-utilized servers. These ratios can be the same in different levels of the cloud system hierarchy or change depending on the aforesaid level. PAR parameters for each level of hierarchy used in periodic optimization procedure is estimated by the PAR value derived in the previous epoch due to VM workload changes. Energy cost is set to 15¢ per KWhr.

B) *Base-line heuristics*

To compare the results of the HRM with previous work, a power and migration-aware VM placement algorithm (periodic optimization) called pMapper [5] is modified and implemented. pMapper borrows FFD heuristics from bin-packing problem to find the amount of resource needed from each server. After this step, VMs are sorted based on their required processing size and assigned to the first available server with the least power to the processing capacity ratio. To avoid high migration cost, VM migrations are sorted based on a metric which is the power decrease to the migration cost. VM migrations are performed based on this ranking metric if their metric is greater than one.

Note that pMapper is a centralized VM placement approach, which does not consider the peak power capacity and CRAC efficiency in its algorithms. We thus modified pMapper to address these two issues as follows: (i) When calculating the ranking metric for servers, the effective power consumption of the server in the previous epoch considering the CRAC efficiency factor is used; (ii) A utilization capacity is considered for each server to decrease the possibility of having power capacity violation. Furthermore, after finalizing the VM assignment, the same reactive optimization procedure that we use for handling any peak power capacity violations is applied. With these two changes, pMapper can serve as a good baseline against the proposed methods in this paper.

Moreover, to show the performance loss of using HRM instead of a centralized periodic optimization algorithm with the same decision criteria, we implemented another centralized periodic resource management algorithm called CRM. In this algorithm, VMs are sorted based on their processing requirement size. Starting from the VM with the biggest resource requirement size, servers are sorted based on their associated VM assignment cost and the VM is assigned to the server with minimum assignment cost. After any assignment, resource, peak power and supply cold air temperature is updated.

Note that, the presented reactive optimization technique is also applied to the case with pMapper and CRM algorithms as periodic optimization technique.

C) *Simulation results*

Total number of active VMs and workload intensity (calculated as the summation of minimum resource requirement of active VMs from a reference server type) in each epoch is shown in Figure 3. During the full-day simulation, nearly 1,000,000 workload changes are generated in our simulator.

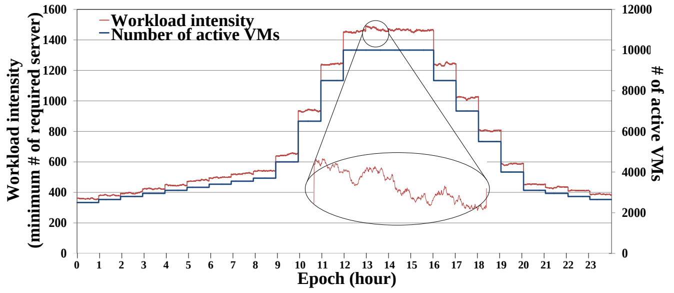


Figure 3- Number of VMs and workload intensity in each epoch

The total costs of the datacenter by applying HRM, CRM or pMapper algorithms are presented in Figure 4.

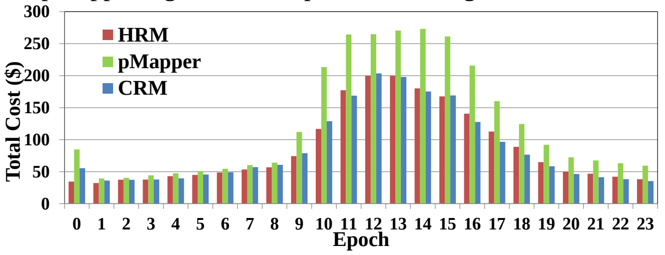


Figure 4- Total cost of datacenter in each epoch

As it can be seen, considering flexible SLA, cooling system efficiency, and peak power capacity in datacenter makes HRM algorithm better than pMapper (by an average of 43%) in terms of the total cost of the system. Moreover, performance loss of HRM method with respect to CRM approach is less than 2% which shows the effectiveness of the hierarchical management. In order to reach the highest level of consolidation in datacenter without increasing the cooling energy cost or SLA violation penalties, a global view of the resource allocation is needed. However, the rather small cost increase in the proposed HRM method versus the CRM method shows that the hierarchical management solution does not result in much of a decrease in the server consolidation factor compared to the centralized solution. This is because PUR parameter favors consolidation in each management level and enables HRM to react to the energy non-proportional behavior of the servers and IT infrastructure in datacenter. The saving in server energy cost plus SLA penalty by only considering VM sizing based on state of the datacenter is around 18% as reported in [4].

One of the key motivations for HRM is to reduce the run-time of the periodic optimization procedures and improve the solution scalability. Run-time of HRM, CRM and pMapper periodic optimization procedures are shown in Figure 5. As it can be seen, the proposed hierarchical method has a very short run-time compared to the centralized approaches. In fact, HRM is in average 7 times faster than pMapper and 27 times faster than CRM.

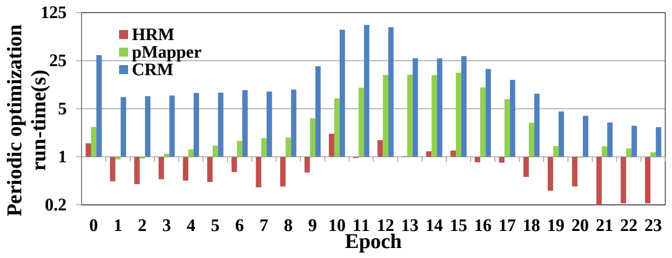


Figure 5- Run-time of the periodic optimization procedure in each epoch (run-time is reported in logarithmic scales)

Calculating the arrival rate for each VM based on the prediction about its workload significantly affects the periodic optimization solution. Moreover, due to migration cost, effect of periodic optimization solution does not disappear even in case of very dynamic workloads. To show this effect, we considered over-provisioning (20% increase in predicted arrival rate) and under-provisioning (20% decrease in predicted arrival rate) with respect to HRM algorithm for calculating the arrival rate. The results of these two scenarios in addition to the HRM results are shown in Figure 6.

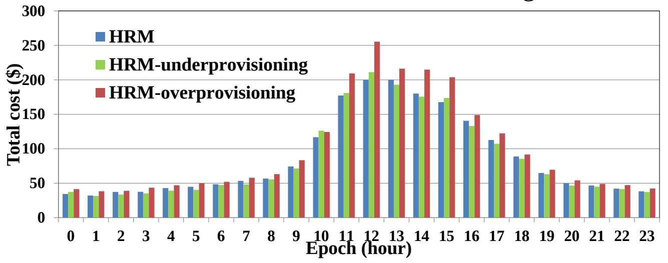
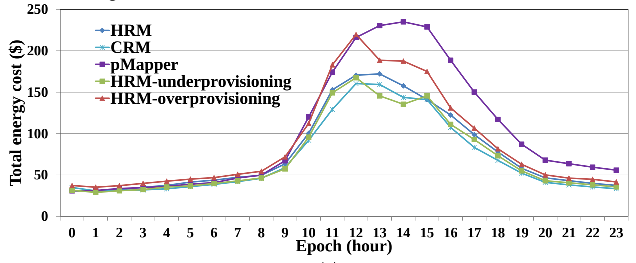


Figure 6- Run-time of the periodic optimization procedure in each epoch

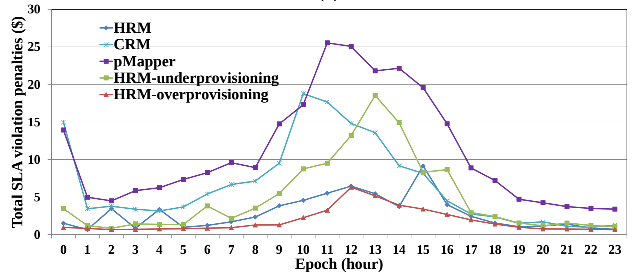
It can be seen that the over-provisioning results in 13% higher cost than HRM and under-provisioning results in 2% lower cost than HRM. Note that, this 2% reduction in total cost results in higher SLA hard constraint violation (low user satisfaction) and significantly higher number of reactive optimization calls in each epoch due to SLA emergencies.

To understand the operational cost of the datacenter, different elements of the total cost having different periodic optimization techniques are shown in Figure 7.

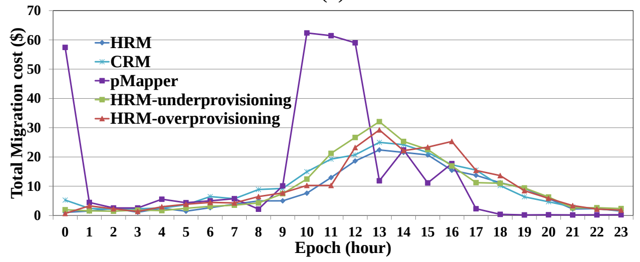
As can be seen, the over-provisioning approach results in high energy cost and low SLA violation penalty and migration cost. This is due to the fact that the amount of resources allocated to each VM is more than what it needs to be. In contrast, the under-provisioning approach results in low energy cost but high SLA violation penalty and migration cost. Moreover, SLA hard constraint violation penalty which results in user dissatisfaction is high in this approach. The pMapper algorithm results in high energy cost and SLA violation penalty due to the lack of flexibility for resource allocation mechanism in this approach but this approach results in the lowest hard SLA constraint violation. Moreover, the pMapper algorithm does not consider the cooling system efficiency and may result in uneven temperature distribution inside containers and higher power usage effectiveness. CRM approach results in lower energy cost with respect to HRM solution but SLA violation penalty, VM migration cost and SLA hard constraint violation penalty are higher in CRM approach. CRM approach results in higher consolidation level than HRM solution due to global search between all servers for each VM assignment in CRM approach which results in lower energy cost and possibly higher resource contention between VMs and higher probability of forced VM migration due to emergencies.



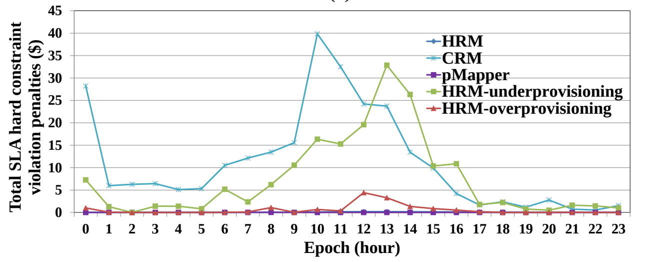
(a)



(b)



(c)



(d)

Figure 7- Elements of the total cost: (a) Energy cost, (b) SLA violation penalty, (c) Migration cost, and, (d) SLA hard constraint violation penalty.

Figure 8 reports the total number of calls to the reactive optimization procedures in one day considering different management strategies.

As expected, the number of calls to reactive optimization procedures for the under-provisioning scenario is larger than other hierarchical management algorithms. Moreover, the rather large number of calls to reactive optimization procedures for pMapper scenario has two important reasons: (i) usual power capacity violation due to issues in periodic solution, and (ii) dynamic calls to solve SLA violation problems due to inflexibility of the resource allocation procedure.

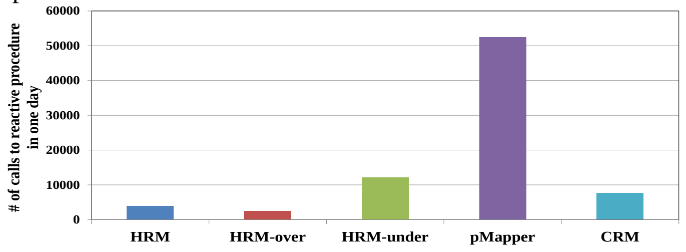


Figure 8- Total number of calls to the reactive optimization procedures in one day

To show the effectiveness of HRM algorithms in reducing the cooling system power consumption, Figure 9 and Figure 10 show two instances of power and temperature distribution in 10-rack containers. In these containers, two rows of racks (five racks in each row) are placed in parallel. Each rack has five chassis (A on bottom to E on top).

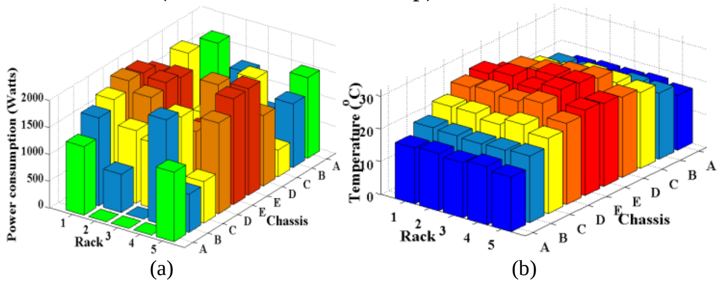


Figure 9- (a) power and (b) temperature distribution in a container with heavy workload. $T_s = 15^oC$ and $max\ T_q^{in} = 30.2^oC$

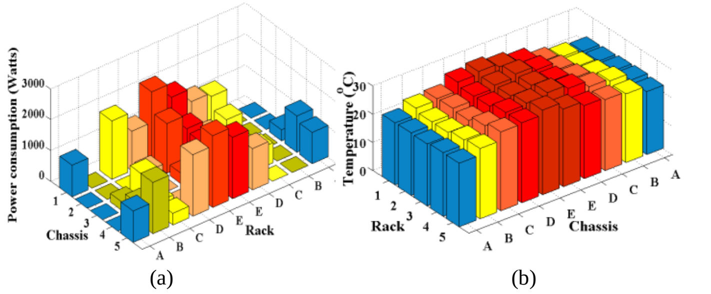


Figure 10- (a) power and (b) temperature distribution in a container with light workload. $T_s = 22^oC$ and $\max T_q^{in} = 29.9^oC$

Figure 9 shows a case with heavier workload having supply cold air temperature equal to 15^{o}C and maximum temperature of 30.2^{o}C. Figure 10 shows a case with lighter workload having supply cold air temperature equal to 22^{o}C and maximum temperature of 29.9^{o}C. In both cases, it can be seen that the resource manager tries to decrease heat recirculation which is the main results of the literature in temperature-aware task scheduling in datacenter such as [17]. Note that this is not always possible because the resource managers also consider three important factors in the resource allocation which are energy proportionality, VM migration cost and peak power cap.

Energy price has a big impact in determining the total cost of the system. Energy price is an important factor in determining the resource allocation strategy in datacenters. All of the proposed algorithms in this paper consider the energy price to decide about the amount of resource allocated to each VM. To examine the effect of energy price on the result of the proposed algorithms, we considered a scenario with 5000 VMs with no significant workload change during the day and different energy prices for each epoch. The energy price and total cost of the datacenter in a full day is shown in Figure 11. The shape of time-of-use dependent energy price in Figure 11 is similar the summer rate of the time-of-use energy pricing for business customers in California shown in reference [40]. As can be seen in this figure, the total cost in datacenter has a similar pattern to the energy price in the system.

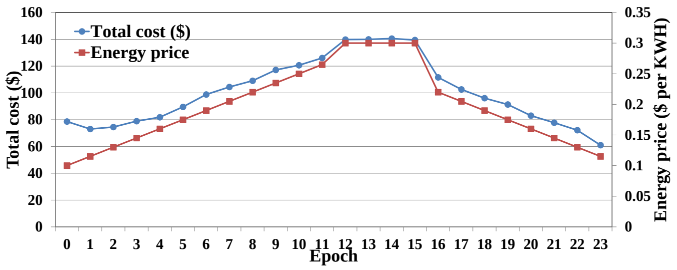
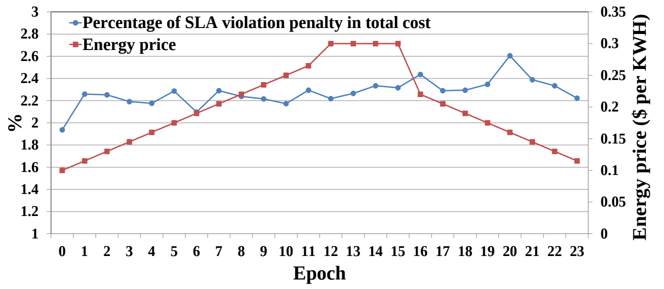
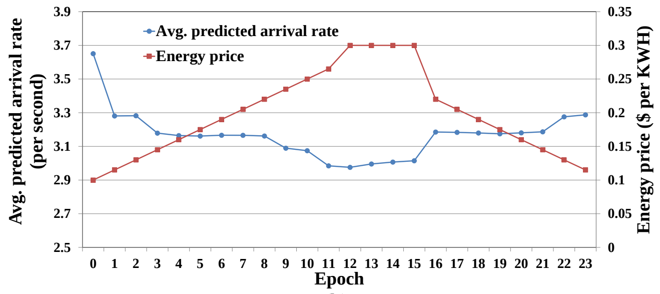


Figure 11- Dynamic energy price and total cost of datacenter in a full day

Figure 12 shows the share of the SLA violation penalty in the total cost and average predicted arrival rate for VMs in different epochs.



(a)



(b)

Figure 12- (a) percentage of the SLA violation penalty in the total cost and (b) average predicted arrival rate for VMs in different epochs

As can be seen in Figure 12(a), the share of SLA violation penalty does not significantly change with energy price. This means that in case of low energy price, resource manager tries to lower the SLA violation penalty and in case of high energy price, it tries to reduce energy consumption by decreasing the resource allocated to each VM. Another proof for this observation is Figure 12(b) which shows that by increasing the energy price, the resource manager decreases the size of each VM (under-provision) to reduce the energy cost.

IX. Conclusion

In this paper, a hierarchical resource management structure for cloud system is presented. The presented structure shows scalability and higher performance compared to a centralized structure suggested in the literature. Adding the flexibility of

SLA-based VM sizing to the resource management problem is a big contributor to the higher performance of the solution compared to the previous approach. Moreover, the performance loss of the decentralized approach with respect to the centralized version of the algorithm is less than 2% having 27 times shorter run-time.

The proposed algorithm results in higher energy proportionality in the overall datacenter, lower SLA violation penalty and migration cost and higher cooling system efficiency. The proposed management structure is appropriate for localized VM migrations and resource allocation adjustment to avoid temperature, peak power and SLA emergencies. The effect of over and under-provisioning approaches in this algorithm is studied in the simulation results.

References

[1] "DatacenterDynamics Global Industry Census 2011," [Online]. Available: http://www.datacenterdynamics.com.br/research/market-growth-2011-2012.

[2] G. Cook, "How clean is your cloud? Catalysing an energy revolution," Greenpeace International, 2012.

[3] L. A. Barroso and U.Hölzle, "The Case for Energy-Proportional Computing," *IEEE Computer*, vol. 40, 2007.

[4] H. Goudarzi, M. Ghasemazar and M. Pedram, "SLA-based Optimization of Power and Migration Cost in Cloud Computing," in *12th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing (CCGrid)*, 2012.

[5] A. Verma, P. Ahuja and A. Neogi, "pMapper: Power and migration cost aware application placement in virtualized systems," in *ACM/IFIP/USENIX 9th International Middleware Conference*, 2008.

[6] Z. Xiao, W. Song and Q. Chen, "Dynamic Resource Allocation Using Virtual Machines for Cloud Computing Environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1107,1117, 2013.

[7] D. Novakovic, N. Vasic, S. Novakovic, D. Kostic and R. Bianchini, "Deepdive: Transparently identifying and managing performance interference in virtualized environments," EPFL, 2013.

[8] M. N. Bennani and D. A. Menasce, "Resource allocation for autonomic datacenters using analytic performance models," in *Second International Conference on Autonomic Computing*, 2005.

[9] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer and A. Tantawi, "An analytical model for multi-tier internet services and its applications," in *SIGMETRICS 2005: International Conference on Measurement and Modeling of Computer Systems*, 2005.

[10] Z. Liu, M. S. Squillante and J. L. Wolf, "On maximizing service-level-agreement profits," in *Third ACM Conference on Electronic Commerce*, 2001.

[11] D. Ardagna, B. Panicucci, M. Trubian and L. Zhang, "Energy-Aware Autonomic Resource Allocation in Multi-Tier Virtualized Environments," *IEEE Transactions on Services Computing*, vol. 99, 2010.

[12] L. Zhang and D. Ardagna, "SLA based profit optimization in autonomic computing systems," in *2nd international conference on Service oriented computing*, 2004.

[13] C.-J. Huang, C.-T. Guan, H.-M. Chen, Y.-W. Wang, S.-C. Chang, C.-Y. Li and C.-H. Weng, "An adaptive resource management scheme in cloud computing," *Engineering Applications of Artificial Intelligence*, vol. 26, no. 1, pp. 382-389, 2013.

[14] A. Gandhi, Y. Chen, D. Gmach, M. Arlitt and M. Marwah, "Minimizing data center sla violations and power consumption via hybrid resource provisioning," in *International Green Computing Conference and Workshops (IGCC)*, 2011.

[15] Q. Tang, S. Gupta and G. Varsamopoulos, "Energy-Efficient Thermal-Aware Task Scheduling for Homogeneous High-Performance Computing Datacenters: A Cyber-Physical Approach," *IEEE Transactions on Parallel and Distributed Systems*, 2008.

[16] R. K. Sharma, C. E. Bash, C. D. Patel, R. J. Friedrich and J. S. & Chase, "Balance of power: Dynamic thermal management for internet data centers," *IEEE Internet Computing*, vol. 9, no. 1, 2005.

[17] Q. Tang, S. Gupta and G. Varsamopoulos, "Thermal-Aware Task Scheduling for Datacenters through Minimizing Heat Recirculation," in *Proc. IEEE Cluster*, 2007.

[18] M. Polverini, A. Cianfrani, S. Ren and A. Vasilakos, "Thermal-Aware Scheduling of Batch Jobs in Geographically Distributed Data Centers," *IEEE Transactions on Cloud Computing*, vol. 2, no. 1, 2014.

[19] R. Raghavendra, P. Ranganathan, V. Talwar, Z. Wang and X. Zhu, "No "power" struggles: Coordinated multi-level power management for the datacenter," *ACM SIGPLAN Notices*, vol. 43, no. 3, pp. 48-59, 2008.

[20] X. Fan, W. Weber and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *in Proceedings of the 34th Annual International symposium on Computer Architecture*, San Diego, CA, 2007.

[21] D. Meisner, C. Sadler, L. Barroso, W. Weber and T. Wenisch, "Power Management of Online Data-Intensive Services," in *Proceedings of the 38th Annual International symposium on Computer Architecture*, 2011.

[22] S. Pelley, D. Meisner, P. Zandevakili, T. F. Wenisch and J. Underwood, "Power Routing : Dynamic Power Provisioning in the Datacenter," in *ASPLOS '10: Architectural Support for Programming Languages and Operating Systems*, 2010.

[23] A. Gandhi, M. Harchol-Balter, R. Das and C. Lefurgy, "Optimal power allocation in server farms," in *international joint conference on Measurement and modeling of computer systems (SIGMETRICS '09)*, 2009.

[24] W. Felter, K. Rajamani, T. Keller and C. Rusu, "A performance-conserving approach for reducing peak power consumption in server systems," in *19th annual international conference on Supercomputing (ICS '05)*, 2005.

[25] B. Addis, D. Ardagna, B. Panicucci, M. Squillante and L. Zhang, "A Hierarchical Approach for the Resource Management of Very Large Cloud Platforms," *IEEE Transactions on Dependable and Secure Computing*, vol. 10, no. 5, pp. 253-272, 2013.

[26] E. Feller, C. Rohr, D. Margery and C. Morin, "Energy Management in IaaS Clouds: A Holistic Approach," in *Proceedings of IEEE 5th International Conference on Cloud Computing (CLOUD)*, 2012.

[27] C. Mastroianni, M. Meo and G. Papuzzo, "Probabilistic Consolidation of Virtual Machines in Self-Organizing Cloud Data Centers," *IEEE Transactions on Cloud Computing*, vol. 1, no. 2, 2013.

[28] N. Tolia, Z. Wang, M. Marwah, C. Bash, P. Ranganathan and X. & Zhu, "Delivering Energy Proportionality with Non Energy-Proportional Systems-Optimizing the Ensemble," in *HotPower*, 2008.

[29] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang and N. Gautam, "Managing server energy and operational costs in hosting centers," in *ACM SIGMETRICS '05*, 2005.

[30] X. Wang and M. Chen, "Cluster-level feedback power control for performance optimization," in *IEEE HPCA*, 2008.

[31] X. Wang and Y. Wang, "Co-con: Coordinated control of power and application performance for virtualized server clusters," in *IEEE 17th International workhop on Quality of Service (IWQoS)*, 2009.

[32] L. A. Barroso and U. Holzle, The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Morgan & Claypool Publishers, 2009.

[33] A. Qouneh, C. Li and T. Li, "A quantitative analysis of cooling power in container-based data centers," in *IEEE International Symposium on Workload Characterization (IISWC)*, 2011.

[34] N. Rasmussen, "Calculating Total Cooling Requirements for Datacenters," *American Power Conversion*, 2007.

[35] C. Tang, M. Steinder, M. Spreitzer and G. Pacifici, "A scalable application placement controller for enterprise datacenters," in *16th International World Wide Web Conference, WWW2007*, 2007.

[36] Z. Liu, M. Lin, A. Wierman, S. H. Low and L. L. H. Andrew, "Greening geographical load balancing," in *Proc. ACM SIGMETRICS*, San Jose, CA, 2011.

[37] H. Goudarzi, "SLA-based, Energy-Efficient Resource Management in Cloud Computing Systems," UNIVERSITY OF SOUTHERN CALIFORNIA, 2013.

[38] "http://ark.intel.com/," [Online].

[39] "http://aws.amazon.com/ec2/#pricing," [Online].

[40] "http://www.pge.com/en/mybusiness/rates/tvp/toupricing.page," [Online].

Hadi Goudarzi received the B.Sc. and M.Sc. degree from the Sharif University of Technology, Tehran, Iran, in 2006 and 2008, respectively, both in electrical engineering (communications), and the Ph.D. degree in electrical engineering from the University of Southern California in 2013. Since 2013, he has been with Qualcomm, Inc., San Diego, CA, USA, as a Senior Engineer. His current research interests include energy-efficient computing, system-level low-power design, and optimization.

Massoud Pedram (M'91–F'00) received the B.S. degree in electrical engineering from the California Institute of Technology, Pasadena, in 1986, and the M.S. and Ph.D. degrees in electrical engineering and computer sciences from the University of California, Berkeley, in 1989 and 1991, respectively. Then, he joined the Department of Electrical Engineering–Systems, University of Southern California, Los Angeles, where he is currently a Pro- fessor and the Chair of computer engineering. He has published four books and more than 500 journal and conference papers. His current research interests include energy-efficient computing, energy storage systems, low-power electronics and design, and computer-aided design of very large scale integration circuits and systems. Dr. Pedram was a recipient of the NSF's Young Investigator Award in 1994 and the Presidential Faculty Fellows Award (i.e., PECASE Award) in 1996. He is an ACM Distinguished Scientist.