Temperature-Aware Dynamic Resource Provisioning in a Power-Optimized Datacenter

Ehsan Pakbaznia, Mohammad Ghasemazar, and Massoud Pedram
University of Southern California
Department of Electrical Engineering-Systems
Los Angeles, U.S.A.
{pakbazni, ghasemaz, pedram}@usc.edu

Abstract - **The current energy and environmental cost trends of datacenters are unsustainable. It is critically important to develop datacenter-wide power and thermal management (PTM) solutions that improve the energy efficiency of the datacenters. This paper describes one such approach where a PTM engine decides on the number and placement of ON servers while simultaneously adjusting the supplied cold air temperature. The goal is to minimize the total power consumption (for both servers and air conditioning units) while meeting an upper bound on the maximum temperature seen in any server chassis in the data center. To achieve this goal, it is important to be able to predict the incoming workload in terms of requests per second (which is done by using a short-term workload forecasting technique) and to have efficient runtime policies for bringing new servers online when the workload is high or shutting them off when the workload is low. Datacenter-wide power saving is thus achieved by a combination of chassis consolidation and efficient cooling. Experimental results demonstrate the effectiveness of the proposed dynamic resource provisioning method.** [1]

Keywords-datacenter, cloud computing, resource provisioning, energy efficient, power optimization, temperature aware

I. INTRODUCTION

Cloud computing services provided by big players such as Amazon, Google, Microsoft, Yahoo, etc are giving a dramatic rise in the Internet-based applications and services. Cloud computing referees to the *applications* and *platforms* delivered as services to different range of clients over the Internet and the *datacenter infrastructure* that provides those services [1]. The cloud itself refers to the datacenter hardware and software infrastructure. Public clouds provide *utility computing* services and are available to the public in the "pay-as-you-go" manner. Cloud computing comprises of infrastructure (datacenter facility), platforms, and applications with applications considered to be the end product and platforms to facilitate the environment to create the applications. Platforms and applications are usually provided as Platform as a Service (PaaS) and Software as a Service (SaaS), respectively.

Datacenters provide the supporting infrastructure for a wide range of applications and services including social and business networking, Webmail, Web search, electronic funds transfer, supply chain management, Internet marketing, online transaction processing, automated data collection systems, High Performance Computing (HPC), etc. The increasing demand for Internet-based services has made the datacenter facilities to grow rapidly. The continuous increase in computing and storage capacities of datacenters is made possible by advances in the underlying manufacturing process and design technologies. A by-product of such a capacity growth has been a rapid rise in the energy consumption and power density of datacenters. However, the continued growth of datacenters is now hindered by their unsustainable (and rising) energy needs. Apart from datacenter energy consumption and associated costs, corporations and governments are also concerned about the environmental impact of datacenters, in terms of their carbon dioxide (CO_2) footprint. Motivated by the need for datacenters to be put on a more scalable and sustainable energy-efficiency curve, this paper seeks to advance the technology of energy-efficient datacenters.

There are a number of different techniques to reduce the energy cost and power density in datacenters in different levels of granularity, chip-level, server level, rack level, datacenter level, etc. There are several works published recently addressing some of the chip-level power optimization issues [2][3]. Load balancing [4][5][6] which is a datacenter-level approach can be used to distribute the total workload of the datacenter among different servers in order to balance the per-server workload (and hence achieve uniform power density). Server *consolidation* [7], which refers to using the minimum number of active servers in the datacenter, is another approach for power reduction of datacenters.

Accounting for about 30% of the total energy cost of a datacenter (another 10-15% is due to power distribution and conversion losses in the datacenter), the cooling cost is one of the major contributors of the total electricity bill of large datacenters [8]. There have been a number of prior works on increasing the efficiency of the cooling process in datacenters by performing temperature-aware task placement [9][10]. In [10] the authors formulate and solve a mathematical problem that maximizes the steady state datacenter cooling efficiency by maximizing the required supplied cold-air temperature value. We used a combination of chassis consolidation and efficient cooling in [11] to minimize the total datacenter power consumption (server plus cooling) and showed that the maximum cooling efficiency does not necessarily result in minimum total datacenter power consumption.

In this paper, we present a power and thermal management (PTM) framework for production datacenters where the (server) resources are dynamically provisioned to meet the required workload while ensuring that a maximum temperature threshold is met throughout the datacenter. The goal is to minimize the total power consumption of the datacenter including the power consumed by the servers and the air conditioning units. We do not explicitly address the power consumed by the network cards and switching gear within the datacenter here. Two actions are taken by the PTM. First is to determine the number of required servers by employing a short-term forecasting technique to predict the datacenter workload. Second is to optimally choose servers that are either being *retired* or *employed* from the available pool of servers and to determine the optimum supplied cold-air temperature value of the AC unit while satisfying the datacenter thermal constraints. The terms retired and employed servers refer to servers that are being turned OFF or ON, respectively. The power saving is thus achieved by a combination of chassis consolidation and efficient cooling.

[1] This work was supported in part by a grant from NSF, Computer Systems Research program.

II. PRELIMINARIES

In this section we give an overview of the datacenter layout, arrangement of servers, the cooling system, datacenter power model, and thermodynamic equations for thermal distribution.

A. DATACENTER CONFIGURATION

A datacenter is typically a (warehouse-sized) room with several rows of server cabinets. Each row comprises of several racks (cabinets), each rack contains several chassis, and each chassis contains several (blade) servers. All the blade servers in a chassis share a single power unit of the chassis. A modern datacenter is designed in hot-aisle/cold-aisle style as depicted in Figure 1, where each row is sandwiched between a hot aisle and a cold aisle. Cold air in cold aisles is supplied by the AC unit and comes through the perforated tiles in the floor. Servers suck the cold air coming from the cold aisle into the rack using chassis fans. The cold air cools the servers; the hot air exits the rack toward the adjacent hot aisles, and is then extracted from the room by the AC intakes on the ceiling above the hot aisles.

A datacenter may include different classes of servers with different power/performance characteristics which are designed for different purposes. For example, Google search engine contains different classes of servers: web servers, index servers, document servers, etc. [12]. In an optimally designed Google cluster, index servers which are responsible for finding the search query in their indexed data usually run CPU-intensive tasks and thus must comprise high speed CPUs. Document servers which are responsible for loading part of a document from the Google storage do not need a high speed CPU since the tasks they run are not CPU intensive.

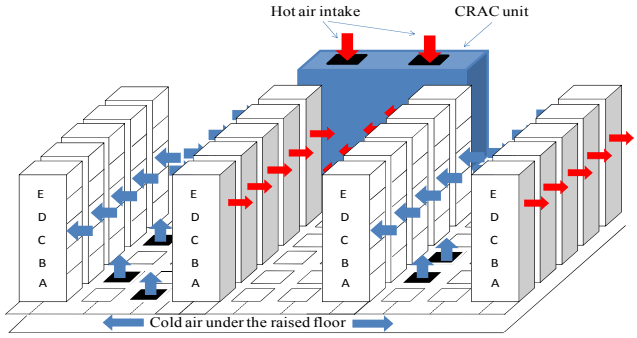


Figure 1. Hot-aisle/cold-aisle datacenter structure.

B. POWER MODEL FOR BLADE SERVERS

Assume there are K different classes of servers distributed among N chassis in the datacenter with the i^{th} chassis containing M_{ij} number of *type-j* servers. Each chassis contains a fixed number of servers, $M = \sum_{j=1}^{K} M_{ij}$. Let c_{ij} denote the number of ON *type-j* servers in the i^{th} chassis. The power consumption of this chassis is calculated as:

$$p_i = \gamma_i + \sum_{j=1}^{K} \alpha_j c_{ij} \qquad (1)$$

where γ_i represents the *base* power consumption of the i^{th} chassis, and accounts for the power consumption of the chassis fan and switching losses due to AC-DC conversion. α_j denotes the power consumption of a *type-j* server when it is ON. We define $\boldsymbol{\gamma} = [\gamma_i]_{N\times 1}$ and $\boldsymbol{\alpha} = [\alpha_j]_{K\times 1}$ as vectors representing base power dissipations of all chassis, and power dissipations of different server classes, respectively. Also, $\boldsymbol{C} = [c_{ij}]_{N\times K}$ denotes the *server state matrix* where c_{ij} is the number of ON *type-j* servers on the i^{th} chassis. We write (1) in matrix form as:

$$\boldsymbol{p} = \boldsymbol{\gamma} + \boldsymbol{C\alpha} \qquad (2)$$

where $\boldsymbol{p} = [p_i]_{N\times 1}$. Chassis base power consumption is typically very high; hence, it is desirable to have the required number of ON servers on the minimum number of chassis so that the remaining ones can be off. This is called *chassis consolidation.*

C. HEAT TRANSFER EQUATIONS

The temperature spatial granularity considered in this paper is at the chassis level. The temperature of the cold air that is drawn to the i^{th} chassis is called *inlet temperature* of that chassis and is denoted by T_{in}^{i}. Similarly, the *outlet temperature* of the i^{th} chassis, T_{out}^{i}, is defined as temperature of the hot air that exits the chassis. The inlet temperature of a chassis depends on the supplied cold air temperature from the Computer Room Air Conditioning (CRAC) unit and the hot air that is *re-circulated* from the outlet of other chassis. The authors in [10] showed that the recirculation of heat in a datacenter can be described by a cross-interference matrix. This matrix is represented by $\boldsymbol{\Phi} = [\phi_{ij}]_{N\times N}$ and shows how much of the inlet heat rate of each chassis comes from the outlet heat rate of other chassis resulting in:

$$\boldsymbol{t_{in}} = \boldsymbol{t_s} + \boldsymbol{Dp}, \quad \boldsymbol{D} = [(\boldsymbol{K} - \boldsymbol{\Phi}^T\boldsymbol{K})^{-1} - \boldsymbol{K}^{-1}] \qquad (3)$$

where $\boldsymbol{T_{in}}$ and $\boldsymbol{T_s}$ are the corresponding inlet temperature and the cold air supply vectors, respectively, and $\boldsymbol{K}$ is an N×N diagonal matrix whose entries are the thermodynamic constants of different chassis, i.e., $\boldsymbol{K} = diag(K_1, \dots, K_N)$, and $K_i = \rho f_i c_p$. It is clear from (3) that the power distribution among different chassis in the datacenter directly affects the temperature distribution in the room. If we use equation (2) to substitute $\boldsymbol{P}$ into (3), we have:

$$\boldsymbol{T_{in}} = \boldsymbol{T_s} + \boldsymbol{D}(\boldsymbol{\gamma} + \boldsymbol{C\alpha}) \qquad (4)$$

III. DATACENTER POWER MODELING

A. POWER CONSUMPTION OF THE CRAC UNIT

The efficiency of the cooling process depends on different factors such as the substance used in the chiller, the speed of the air exiting the CRAC unit, etc. *Coefficient of Performance* (COP) is defined as the ratio of the amount of heat that is removed by the CRAC unit (Q) to the total amount of energy that is consumed in the CRAC unit to chill the air (E) [9]:

$$COP = Q/E \qquad (5)$$

The COP of a CRAC unit is not constant and varies by the temperature of the cold air that it supplies to the room. In particular the higher the supplied air temperature, the better cooling efficiency. In this paper we use the COP model of a typical water-chilled CRAC unit utilized in a HP Utility Datacenter [9]. This model is quantified in terms of the supplied cold air temperature (T_s) as follows [9]:

$$COP(T_s) = (0.0068\, T_s^2 + 0.0008\, T_s + 0.458) \qquad (6)$$

B. TOTAL POWER CONSUMPTION

We define the total power consumption of a datacenter as the power consumptions of all chassis and the CRAC unit i.e., we do not consider power losses in the electrical power conversion network (UPS, AC-DC and DC-DC converters) and losses in the switch gear and conductors. The IT power consumption of a datacenter is denoted by P_{IT} and is the summation of power consumption over all chassis:

$$p_{IT} = \sum_{i=1}^{N} p_i \qquad (7)$$

where p_i is the power consumption in the i^{th} chassis. The power cost of the CRAC unit is specified as $p_{CRAC} = p_{IT}/COP(T_s)$. The total datacenter power is the summation of P_{IT} and P_{CRAC} and is written as:

$$p_{DC} = \left(1 + \frac{1}{COP(T_s)}\right) \sum_{i=1}^{N} p_i \qquad (8)$$

Substituting the expression from (1) for p_i, we obtain:

$$p_{DC} = \left(1 + \frac{1}{COP(T_s)}\right) \left(\sum_{i=1}^{N} \gamma_i + \sum_{i=1}^{N}\sum_{j=1}^{K} \alpha_j c_{ij}\right)$$

IV. TEMPERATURE-AWARE DYNAMIC RESOURCE PROVISIONING AND POWER OPTIMIZATION

Figure 2 shows the block diagram of the datacenter management system considered in this paper. Input requests are collected in a Global input Queue (GQ). The Temperature-Aware Dynamic Resource Provisioning (TA-DRP) module consists of two sub-modules: Workload Monitoring (WM) unit and Power-Thermal Manager (PTM) unit. WM does workload analysis and prediction for the next epoch. The result is passed on to the PTM unit in the form of the required number of ON servers for each server class for the next epoch. PTM then uses this number along with information about the servers' status in the datacenter to decide on which servers/chassis to employ/retire (turn ON/OFF) for the next epoch. The Request Dispatcher (RD) unit uses the server status information to assign requests to different servers. Designing a power efficient RD unit is not the purpose of this paper, and when needed we use a simple Round Robin scheduling algorithm. The combination of the WM and the PTM units is called TA-DRP, the main focus of this paper.

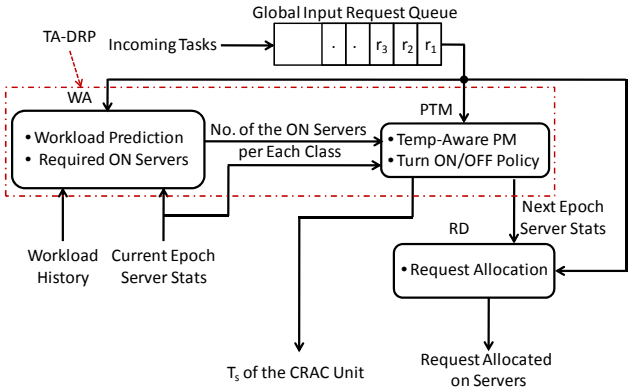


Figure 2. Datacenter power optimization architecture.

A. WORKLOAD MONITOR

As mentioned earlier, WM is responsible for providing PTM with the required number of ON servers for each class. We denote the required number of *type-j* ON servers for the next epoch by $n_j(t+1)$, where the time index "$t+1$" represents the next epoch. The total number of *type-j* servers which must be turned ON in the next epoch is $S_j = n_j(t+1) - n_j(t)$, where the time index "t" represents the current epoch. If $S_j > 0$, the PTM must employ S_j new type-*j* servers; if $S_j < 0$, PTM retires $|S_j|$ type-*j* servers, and if $S_j = 0$, PTM does not take any action for the type-*j* servers.

In this paper we estimate the required number of servers for each epoch by performing workload prediction. To introduce the prediction approach, we need to define two parameters that determine the characteristics of a workload: total number of requests that are being processed at any given time and the request arrival rate. We denote the total number of requests and the request arrival rate at time t by $r(t)$ and $\lambda(t)$, respectively.

Without loss of generality suppose that each request requires n_j^{avg} number of *type-j* servers on average. The total number of required *type-j* servers at time t, and the rate at which this number changes, can be estimated as $r(t)\times n_j^{avg}$ and $\lambda(t)\times n_j^{avg}$, respectively. This is a reasonable assumption because most of the cloud computing services need a relatively fixed number of servers of each type to serve a request. Examples of cloud computing applications include Web services such as Web search, Web mail, Connection services (e.g., Yahoo Messenger, Google Talk, and Windows Live Messenger), and web crawlers, etc. However, these applications demand non-uniform compute resources over time (across multiple decision epochs). Therefore, value of n_j^{avg} is updated by using a moving average.

B. CALCULATING THE REQUIRED SERVER COUNT

For a given maximum tolerable CPU (or I/O) utilization and a specific application workload, we can find a maximum tolerable load (i.e., the number of connections to each server for connection-intensive internet services) and the maximum tolerable rate at which the load for a server is changing [13]. We denote the maximum tolerable load and maximum tolerable load rate for each *type-j* server with R_j^{max} and Λ_j^{max}, respectively. Therefore, our algorithms have to guarantee that the load of any *type-j* server will not exceed R_j^{max}, and the rate at which this load is changing does not exceed Λ_j^{max}. R_j^{max} and Λ_j^{max} depend on the type of application and the amount of bandwidth that the corresponding tasks take of each server. In this paper we assume $R_j^{max}=10$ and $\Lambda_j^{max}=3$.

We may thus calculate the required number of *type-j* servers in the datacenter at time t as [13]:

$$n_j(t) = \max\left\{\left\lceil \gamma_r \frac{r(t)n_j^{avg}}{R_j^{max}} \right\rceil, \left\lceil \gamma_\lambda \frac{\lambda(t)n_j^{avg}}{\Lambda_j^{max}} \right\rceil\right\} \qquad (9)$$

where $\lceil x \rceil$ denotes the ceiling of x, and γ_r and γ_λ are the correction coefficients, and we have $\gamma_r, \gamma_\lambda > 1$. In Section C we explain how these coefficients are chosen. However, as stated in [13] there is a problem with this equation. In (9) we assume that a newly employed server will have R_j^{max} number of connections right after it is turned ON. This is not a valid assumption, and the load of a newly employed server will rise gradually from 0 to R_j^{max}.

C. WORKLOAD PREDICTION

Enterprise datacenters workloads typically show a repetitive pattern with a period in the order of hours, days, weeks and so forth. In [14] authors have demonstrated that for the purpose of workload forecasting, the period of workload behavior is equal to 7 days for a large variety of datacenter applications. The forecasting method we use is composed of two exponential smoothing components for the trend value and the offset value prediction.

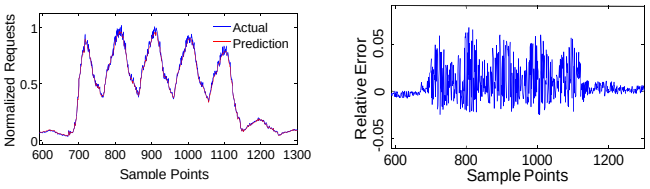
$$s(t) = s_{trend}(t) + s_{offset}(t) \qquad (10)$$

The trend component performs prediction of the periodic pattern that has been exhibited with period T, whereas the offset component uses the correlation between the estimated value and the previous immediate neighbors. In this paper we use the same idea in the form of the following forecasting equation:

$$s(t) = \sum p_i \cdot x(t - iT) + \sum q_i \cdot \big(x(t-1) - x(t-1-iT)\big) \qquad (11)$$

where $x(.)$ and $s(.)$ represent actual and predicted values, respectively. Our experiments show that four p_i coefficients (p_1-p_4) and two q_i coefficients (q_1-q_2) results in a small amount of prediction error. It is worth mentioning that p_i and q_i coefficients are updated adaptively to reflect the time varying behavior of the workload. The assumption is that values of $x(.)$ at every T steps are highly correlated. The offset component reflects the correlation between the observed value differences in the recent history with respect to the predicted trend .

Forecasting $r(t)$ and $\lambda(t)$ is done through equation (11). Our experiments show that four p_i coefficients (p_1-p_4) and two q_i coefficients (q_1-q_2) results in a small amount of prediction error. The predicted values of $r(t)$ and $\lambda(t)$ obtained from (11) are then used to estimate the total number of ON servers using (9). The correction coefficients, γ_r and γ_λ in (9), are calculated based on the real-time prediction error measurements to avoid performance loss. We set the correction factors in (9) to $\gamma_r = 1+3\sigma_r$ and $\gamma_\lambda = 1+3\sigma_\lambda$, where σ_r and σ_λ are the standard deviation of the prediction error of $r(t)$ and $\lambda(t)$, respectively [13]. Figure 3.a and Figure 3.b illustrate the result of the workload forecasting for $r(t)$ and the prediction error in one-week.



(a) Predicted data vs. Actual Data (b) Prediction error

Figure 3. Prediction of total number of requests, r(t).

D. *Power/Thermal Manager*

In this section we present our Power Thermal Manager (PTM) unit which sits at the heart of TA-DRP. As explained in the beginning of Section IV, inputs to the PTM are the server status in the current epoch and the required number of ON servers (of each server class) for the next epoch. Using these inputs, PTM decides on the location of the servers that are being employed or retired (this is sometimes referred to as the server placement problem – the goal is to assign virtual servers or applications to the physical servers in the datacenter so that the spatial distribution of ON and OFF servers result in minimum overall power cost in the datacenter including the power dissipations of the servers and the air conditioning unit). The optimality is defined as minimizing the total datacenter power consumption given in (8).

The goal of PTM unit is to minimize the total datacenter power consumption given in (8) by dynamically employing or retiring the requested number of servers provided by the WM. This is done by a combination of three means: (i) employing the right number of servers of each type for each epoch and retiring any unused servers; (ii) chassis consolidation, i.e., turning ON only the minimum number of chassis and thereby eliminating the unnecessary base power consumption of the chassis, and (iii) maximizing the required T_s, thus a more efficient cooling, by optimally choosing locations of servers and chassis that are to be employed or retired. Outputs of the PTM unit are the supplied cold air temperature value and the exact ON/OFF status of servers/chassis for the next epoch. Every time there is a need to employ new servers, we keep the currently ON servers ON and simply add new servers (we do not retire an ON server and employ a different new server instead.) This is to avoid the performance and energy overheads associated with retiring a busy server, employing a new server, and transferring the retired server's jobs to the new server. The downside of this policy is that it does not guarantee a power optimal solution across all datacenter utilization levels because of the ON server persistency policy.

To state the PTM problem, we first pay attention to the cost function (P_{DC}) given in (8). For simplicity, we extract the T_s dependency from the cost function. The optimum value of T_s will be determined by performing a linear search across all possible T_s values, and finding a value that results in the minimum P_{DC}. Note that for a fixed T_s value, $COP(T_s)$ becomes a constant and can be taken out of the cost function. Then the cost function simply becomes P_{IT}. We introduce a new integer variable for each chassis that takes on values from {0,1} and signals whether a chassis is ON or OFF. This variable is denoted by x_i for the i^{th} chassis, and is defined as:

$$x_i = \begin{cases} 0 \ ; & \sum_{j=1}^{K} c_{ij} = 0 \\ 1 \ ; & \sum_{j=1}^{K} c_{ij} \neq 0 \end{cases} \qquad (12)$$

It can be shown that the cost function (P_{IT}) becomes:

$$\text{Minimize}\{\boldsymbol{\gamma}^T \boldsymbol{x} + \boldsymbol{C\alpha}\mathbf{1}_{1\times N}\} \qquad (13)$$

where $\boldsymbol{x}=[x_1,x_2,\ldots,x_N]^T$, and $\mathbf{1}_{1\times N}$ denotes an N-dimensional row vector with all elements equal to 1. Also, the inlet temperature vector in (4) will change to:

$$\boldsymbol{T}_{in} = \boldsymbol{T}_s + \boldsymbol{D}(\boldsymbol{\Gamma x} + \boldsymbol{C\alpha}) \qquad (14)$$

where $\boldsymbol{\Gamma}$ is a diagonal matrix defined as $\boldsymbol{\Gamma} = diag(\gamma_1, \gamma_2, \ldots, \gamma_N)$. PTM decisions depend on the current server/chassis ON/OFF status. To capture the ON/OFF status of chassis, we define a new column vector, $\boldsymbol{x^0}$, of size N as $\boldsymbol{x^0}=[x_1{}^0, x_2{}^0, \ldots, x_N{}^0]^T$, where $x_i{}^0=1$ if the i^{th} chassis is currently ON, otherwise $x_i{}^0=0$. Similarly we define a new matrix, $\boldsymbol{C}^0 = [c_{ij}^0]_{N\times K}$, to capture the ON/OFF status of each specific type of server. c_{ij}^0 is the number of *type-j* servers that are currently employed on the i^{th} chassis.

Server Retirement Policy

The purpose of the retirement policy is to minimize the number of ON servers by retiring additional ones. Every time the WM decides on reducing the number of ON servers of a certain type, the PTM unit selects some candidate servers to retire. Unlike the turn ON scenario (c.f. Section 0) where the candidate ON servers will be employed immediately, for the turn OFF case, the PTM passes the list of candidate servers to the Request Dispatcher (RD), and asks RD to retire them by simply not assigning new requests to them. On the other hand, the PTM has a list of retiring servers which it updates at each epoch. If a server on this list stays idle, i.e., it does not provide service to any request, for a certain period of time, that server will be put in the halt (hibernate) mode. Note that we do not completely turn OFF the retired servers (unless the whole chassis is being turned OFF), and we put them in hibernate mode instead. This is due to the small amount of power consumption and faster (compared to an OFF server) wakeup time. Similar to the retiring server list, the PTM also maintains a list of retiring chassis comprising ON chassis that include hibernated servers and no ON servers.

Now we explain how the list of candidate retiring servers is determined by the PTM. The power thermal optimization problem to determine the retiring *type-j* servers can be formulated as the following *Integer Linear Programming* (ILP) problem.

$$\text{Minimize}\{\boldsymbol{\gamma}^T \boldsymbol{x} + \boldsymbol{C\alpha}\mathbf{1}_{1\times N}\}$$

s.t.

1. $\boldsymbol{T}_s + \boldsymbol{D}(\boldsymbol{\Gamma x} + \boldsymbol{C\alpha}) \leq \boldsymbol{T}_{critical}$
2. $\sum_{i=1}^{N} c_{ij} = \sum_{i=1}^{N} c_{ij}^0 + S_j \quad ; \forall j \mid S_j < 0$
3. $x_i \leq \sum_{j=1}^{K} c_{ij} \leq \mathrm{M} x_i \quad ; \ 1 \leq i \leq \mathrm{N}$
4. $c_{ij} \leq c_{ij}^0 \quad ; \ 1 \leq i \leq \mathrm{N}$
5. $x_i \in \{0,1\} \quad ; \ 1 \leq i \leq \mathrm{N}$
6. $c_{ij} \in \{0,1,\ldots,\mathrm{M_{ij}}\} \quad ; \ 1 \leq i \leq \mathrm{N}$

(15)

where $\mathbf{T}_{\text{critical}}$ is a vector of size N with all entries equal to a critical inlet temperature, T_{critical} (The inlet temperature of all chassis must be less than this value in order to ensure that the corresponding servers will not overheat and eventually fail). A typical value for T_{critical} is 25°C [10]. The outputs of the ILP problem in (15) are c_{ij} and x_i values. x_i values determine which chassis are to stay ON and which are to be retired. c_{ij} values determine the number of servers from each type that are to be retired on each chassis.

Server Employment Policy

The purpose of the server employment policy is to determine the optimum T_s value and locations of the required number of ON servers, and also to turn them ON. As mentioned in Section 0, the PTM maintains a list of retiring servers and a list of hibernating servers. Each time the PTM is asked to make use of new servers of a certain type, it first tries to meet this request by employing servers from the retiring server list. If this is not possible, then the PTM will try to satisfy the request by employing servers from hibernating

server list. Finally, if this is not possible either, PTM will employ new servers by solving an optimization problem as explained below.

The turn-ON power thermal optimization problem to employ *type-j* servers can be formulated as the following *Integer Linear Programming* (ILP) problem.

$$\text{Minimize}\{\boldsymbol{\gamma}^T\boldsymbol{x} + \boldsymbol{C\alpha}\mathbf{1}_{1\times N}\}$$

s. t.

$$\begin{aligned}
&1.\ \boldsymbol{T}_s + \boldsymbol{D}(\boldsymbol{\Gamma x} + \boldsymbol{C\alpha}) \le \boldsymbol{T}_{critical} \\
&2.\ \textstyle\sum_{i=1}^{N} c_{ij} = \sum_{i=1}^{N} c_{ij}^0 + S_j \quad ; \quad \forall j \mid S_j > 0 \\
&3.\ x_i \le \textstyle\sum_{j=1}^{K} c_{ij} \le \mathrm{M}x_i \quad ; \quad 1 \le i \le \mathrm{N} \\
&4.\ c_{ij}^0 \le c_{ij} \le \mathrm{M_{ij}} \quad ; \quad 1 \le i \le \mathrm{N} \\
&5.\ x_i^0 \le x_i \le 1 \quad ; \quad 1 \le i \le \mathrm{N} \\
&6.\ x_i \in \{0,1\} \quad ; \quad 1 \le i \le \mathrm{N} \\
&7.\ c_{ij} \in \{0,1,\dots,\mathrm{M_{ij}}\} \quad ; \quad 1 \le i \le \mathrm{N}
\end{aligned} \tag{16}$$

Note that the problem statement in (16) is the power optimization problem for a non-idle datacenter (a datacenter that already contains some ON servers.) In that sense it is different from the problem statement presented in [10] which is for an idle datacenter.

Calculating the Optimum T_s value

Every time that we solve the retirement/employment policy, we also need to determine the optimum T_s value. In this paper we perform a linear search on possible T_s values, and solve the retirement/employment problem every time. We then pick the solution that gives the minimum total power consumption.

V. Simulation Results

In this section we evaluate the power dissipation and cooling cost of our proposed technique, and we compare it with some of the common techniques in datacenter operations.

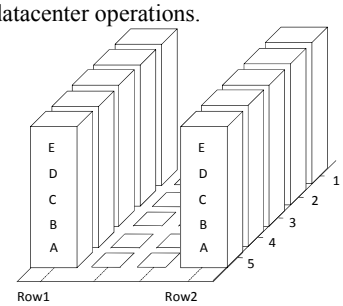


Figure 4. Datacenter structure used in the simulations.

A. Simulation Setup

We use a small scale datacenter with physical dimensions of 9.6m×8.4m×3.6m consisting of 7U blade servers. The datacenter has two rows that are put together in a hot-aisle/cold-aisle arrangement as it is shown in Figure 4. Each row has five 42U racks. Each rack consists of five chassis each having 20 blade servers. Therefore, there are a total number of 1,000 servers in this datacenter. A CRAC unit is used to supply the cold air with f=8m^3/s in the room. We may have K=1 or K=2 type(s) of server in the datacenter. Power parameters for servers and chassis are γ=820W, α_1=85W (uses higher performance core with larger EPI), and α_2=50W (uses lower performance core with smaller EPI). We have simulated the WM and PTM units using the algorithms explained in previous sections. To solve the ILP problems for server employment and retirement, we first used the LP solver package of TOMLAB [15], and then found the closest integer solution to the continuous variable solution.

To the best of our knowledge, the present work is the first that addresses temperature-aware dynamic resource provisioning in a power-optimized datacenter. So, comparison with prior work is not possible. However, we compared the proposed technique (TA-DRP) with two (reasonable) greedy heuristics, called GREEDY and TA-GREEDY. Their difference from TA-DRP is that they use different techniques for server retirement and employment; otherwise they operate with the exact same procedures for the WM and RD units.

Greedy (GREEDY)

This heuristic algorithm performs chassis consolidation using a greedy approach without considering the cooling efficiency factor. For the server employment policy, it starts with chassis that have the maximum occupancy factor (maximum number of employed servers) so that no new chassis is turned on. For the retirement policy, on the other hand, it uses the least occupied chassis, so this chassis will have more chance to be turned off later on when the workload diminishes.

Temperature-Aware Greedy (TA-GREEDY)

In this heuristic algorithm, the chassis' inlet temperatures are given a higher weight (priority) compared to the chassis' occupancy factor. This is to prevent hot spots and imbalanced temperature distribution across the datacenter. Indirectly, balancing the heat distribution in the datacenter can save power. The algorithm maintains a list of relatively hot servers whose inlet temperatures are above a threshold value e.g., T_{th}=22.5°C. These servers will be assigned a higher priority for retirement. If there are no hot servers to retire, this heuristic picks the retiring candidates in the same fashion as GREEDY heuristic. In the same spirit, a cold chassis with maximum number of employed servers will be given a high priority to be used for server employment. Thus TA-GREEDY avoids turning on any servers in a chassis on the hot list as much as possible. Note that both GREEDY and TA-GREEDY adjust the T_s value, if and when needed, so that the thermal constraints are met.

B. Workload Generation

Our simulations were done using a benchmark suite where the number of existing requests and the expected request arrival rate are the input parameters. We could thus simulate a wide range of workload scenarios corresponding to different initial occupancies for the global queue and request arrival rates. The requests were homogenous in terms of their CPU and memory usage.

C. Power and T_S Comparison

Figure 5 shows the total power consumption for the three techniques described above. The figure is the result of running the workload for one full. The workload prediction is done by the WM and the result is passed to PTM. It is seen from Figure 5 that TA-DRP consumes less power than GREEDY and TA-GREEDY.

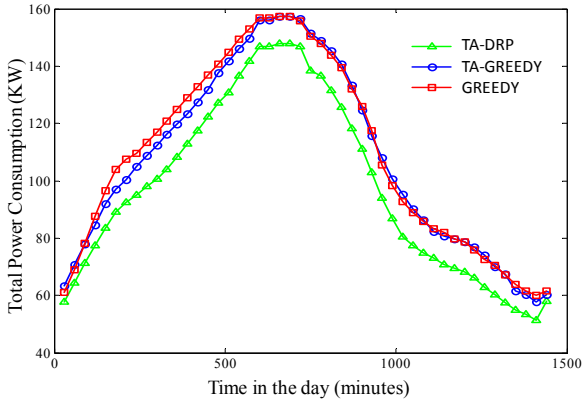
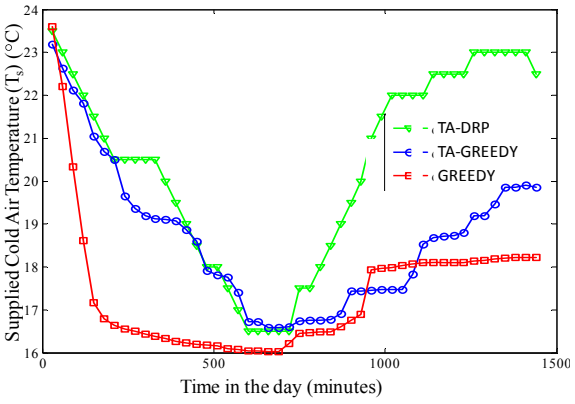


Figure 5. Comparison of the total power consumption for GREEDY, TA-GREEDY and TA-DRP (K=1).

Figure 5 also shows that TA-GREEDY performs better than GREEDY when employing new servers (i.e., when the total power is increasing in the figure). The reason is that TA-GREEDY employs new servers from cool chassis. Therefore, the amount of increase in

the maximum inlet temperature of chassis is very small, and we may only need a small amount of T_s compensation. In contrast, GREEDY employs new servers from the most occupied chassis that are usually the hottest ones with inlet temperature value being very close to the critical temperature. This requires a larger amount of T_s compensation, which results in less efficient cooling and thus larger datacenter power. GREEDY and TA-GREEDY almost perform similar while retiring servers: GREEDY retires servers from the least occupied chassis, but these are usually the coolest chassis i.e., the ones that TA-GREEDY retires.

Figure 6 compares the required T_s values during the day for all the three algorithms discussed in this paper. As it is seen from this figure, the results of TA-GREEDY and TA-DRP are very similar when employing new servers; however, TA-DRP outperforms TA-GREEDY during server retirement. The reason is that TA-GREEDY (unlike TA-DRP) picks hot chassis as the candidate retiring ones. This results in missing opportunities of chassis consolidation and increasing T_s as a reward of that.



D.

Figure 6. T_s comparison (K=1).

Figure 7 shows the power consumption of the datacenter simulated for half a day. In this case all the requests to the datacenter use two different types of servers (K=2). Our future plan is to analyze the characteristics of the workload generated by standard benchmarks such as the SPECpower_ssj2008 [16] and TPC-APP to demonstrate the degree of power savings achieved by TA-DRP for these workloads. Figure 8 shows temperature distribution for two snapshots of the TA-DRP algorithm in the room. In both cases we assume that the all the input requests use two types of servers. Figure 8.a represents the case when 820 servers are ON (410 of each type). TA-DRP has used 40 chassis to provide 820 servers in this case.

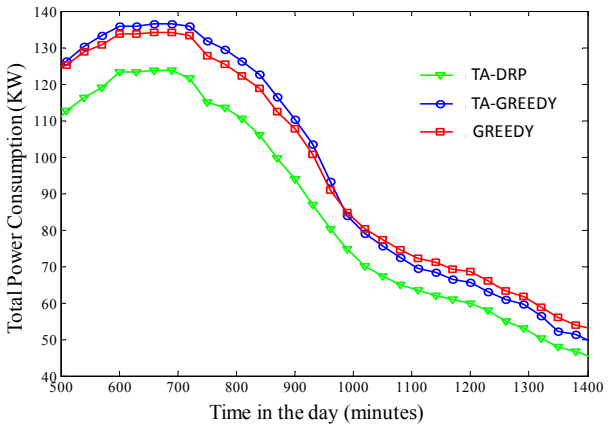
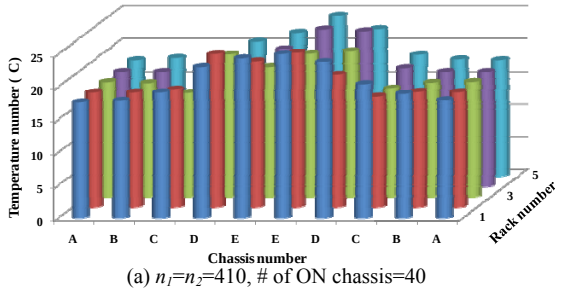


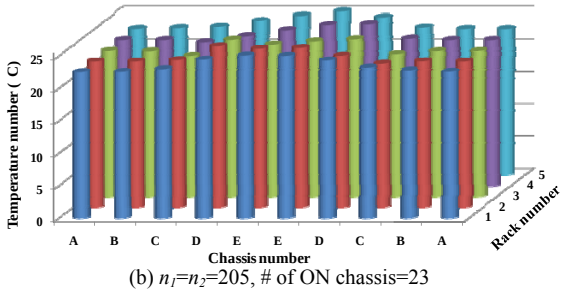
Figure 7. Comparison of the total power consumption for K=2.

VI. CONCLUDING REMARKS

We presented a power-optimized datacenter that performs dynamic provisioning of its cyber-physical resources. Power saving was achieved by a combination of *chassis consolidation* and efficient cooling. Experimental results showed the effectiveness of the proposed dynamic datacenter resource provisioning scheme. Future work will focus on extending this work to include more sophisticated request scheduling algorithm and to do proof-of-concept demonstrations on a small-scale production datacenter at our site.



(a) n_1=n_2=410, # of ON chassis=40



(b) n_1=n_2=205, # of ON chassis=23

Figure 8. Temperature distribution of a snapshot of TA-DRP.

REFERENCES

[1] M. Armbrust, et al., "Above the Clouds: A Berkeley View of Cloud computing," *Technical report*, Univ. of California at Berkeley, 2009.

[2] A. Bogliolo, et al., "Specification and analysis of power-managed systems" *Proc. of the IEEE*, Vol. 92, Issue 8, pp 1308–1346, Aug. 2004.

[3] A. Mutapcic, et al., "Processor Speed Control with Thermal Constraints" *IEEE Trans. on Cir and Sys*, vol. 56, issue 9, pp. 1994–2008, 2009.

[4] V. Cardellini, et al. "Dynamic load balancing on Web-server systems," *IEEE Internet Computing Magazine*, vol. 3, issue 3, pp. 28-39, 1999.

[5] D.M. Dias, et al. "A Scalable and Highly Available Web Server," *Proc. IEEE Computer Soc. Int'l Conf.,* pp. 85–92, Feb. 1996.

[6] E. Pinheiro, et al. "Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems," *Workshop on Compilers and Operating Systems for Low Power,* 2001.

[7] http://www.sun.com/x64/intel/consolidate-using-quadcore.pdf

[8] N. Rasmussen, "Calculating Total Cooling Requirements for Data Centers," American Power Conversion, white paper number 25, 2007.

[9] J. Moore, et al. "Making scheduling 'cool': Temperature-aware resource assignment in data centers," Usenix Annual Technical Conf., 2005.

[10] Q. Tang, et al. "Energy-Efficient Thermal-Aware Task Scheduling for Homogeneous High-Performance Computing Data Centers: A Cyber-Physical Approach," *IEEE Tran. Parallel and Dist. Sys.* 2008.

[11] E. Pakbaznia and M. Pedram, "Minimizing data center cooling and server power costs," *Proc. of International Symposium on Low Power Electronics and Design,* pp. 145-150, Aug. 2009.

[12] L. A. Barroso, et al., "Web search for a planet: The Google cluster architecture," *IEEE Micro*, vol. 23, no. 2, pp. 22-28, April 2003.

[13] G. Chen, et al. "Energy-aware server provisioning and load dispatching for connection-intensive internet services," *Proc. of USENIX Symp. on Networked Systems Design and Implementation*, pp. 337-350, 2008.

[14] D. Gmach, et al., "Workload Analysis and Demand Prediction of Enterprise Data Center Applications," *Proc. of IISWC*, Sep. 2007.

[15] http://tomopt.com/tomlab/

[16] http://www.spec.org/power_ssj2008/