Energy Management in Cross-Domain Content Delivery Networks: A Theoretical Perspective

Chang Ge, *Student Member, IEEE*, Zhili Sun, *Member, IEEE*, Ning Wang, *Member, IEEE*, Ke Xu, *Senior Member, IEEE*, and Jinsong Wu, *Senior Member, IEEE*

***Abstract*—In a content delivery network (CDN), the energy cost is dominated by its geographically distributed data centers (DCs). Generally within a DC, the energy consumption is dominated by its server infrastructure and cooling system, with each contributing approximately half. However, existing research work has been addressing energy efficiency on these two sides separately. In this paper, we *jointly* optimize the energy consumption of both server infrastructures and cooling systems in a holistic manner. Such an objective is achieved through both strategies of: 1) putting idle servers to sleep within individual DCs; and 2) shutting down idle DCs entirely during off-peak hours. Based on these strategies, we develop a heuristic algorithm, which concentrates user request resolution to fewer DCs, so that some DCs may become completely idle and hence have the opportunity to be shut down to reduce their cooling energy consumption. Meanwhile, QoS constraints are respected in the algorithm to assure service availability and end-to-end delay. Through simulations under realistic scenarios, our algorithm is able to achieve an energy-saving gain of up to 62.1% over an existing CDN energy-saving scheme. This result is bound to be near-optimal by our theoretically-derived lower bound on energy-saving performance.**

***Index Terms*—Content delivery network, data center, energy management.**

I. Introduction

GENERALLY, a content delivery network (CDN) operator strategically establishes a number of data centers (DC) at multiple distributed sites towards the edge of the Internet. Web contents are replicated and cached at these DCs, so that end-users can experience reduced end-to-end delay and enhanced service availability when requesting contents. Such an infrastructure typically involves dozens of large DCs (each with thousands of servers) or hundreds to thousands of smaller DCs (each with hundreds of servers), which are geographically distributed across large geographical areas. It is adopted by major CDN operators such as Akamai and Limelight [1]. The contribution of DCs worldwide to the global energy consumption has increased from 1% in 2005 to around 1.5% to 2010, which is expected to grow by 15% every year in the foreseeable future [2]. Therefore, it is crucial to develop effective energy-saving schemes for CDNs.

To reduce CDN energy consumption, the common practice in the literature is to dynamically provision fewer active servers within individual DCs during off-peak hours, so that the idle servers are put into the sleep mode [3]–[5]. Furthermore, energy management schemes of cooling systems in individual DCs have been proposed [6], [7]. Through these strategies, most existing schemes focused on managing energy costs of servers or cooling systems *separately*. Since cooling systems in modern DCs contribute around 44.4% to 47% to the overall energy cost on average [2], [8], to *holistically* maximize energy saving in CDNs, energy consumption of servers and cooling systems need to be optimized simultaneously by taking into account their dependency.

In this paper, our goal is to jointly minimize both server's and cooling system's energy consumption among geographically-distributed DCs. We employ two strategies simultaneously to achieve this goal. Firstly, within each individual DC, server energy consumption is reduced through concentrating loads to fewer servers, so that the remaining idle servers can be put to sleep [3]–[5]. Secondly, among multiple DCs in a CDN, we aim to concentrate request resolution to fewer DCs, so that some DCs without any mapped request can be shut down entirely. Such an idea is based on the fact that an *idle* cooling system typically consumes around 40% of its peak power consumption in an active DC [9], while it consumes zero power in a shut down DC [10]. Furthermore, DC shutdown operations are not uncommon in the DC industry, which are performed mainly for hardware maintenance etc. Therefore, during off-peak hours, it is likely that controlled DC shutdown can produce substantial energy-saving gains over existing server-sleeping-only schemes.

Despite the very promising potential, there are distinct challenges associated with such joint optimization. When managing server sleeping and DC shutdown in a CDN, the trade-off between end-to-end QoS performance and energy-saving gain must be well balanced. Intuitively, after some DCs are shut down, some requests might have to be resolved to alternative DCs in further remote locations. Specifically, in a cross-domain CDN, this might even lead to inter-domain request redirection which would cause poorer service availability and user-perceived end-to-end delay. This challenge can be tackled through strategic planning on user-to-DC request resolution. When resolving end-users' requests to DCs, instead of

Manuscript received May 7, 2014; revised August 1, 2014; accepted August 6, 2014. Date of publication August 13, 2014; date of current version September 5, 2014. This work is supported by EU FP7 EVANS Project (PIRSES-GA-2010-269323). The associate editor coordinating the review of this paper and approving it for publication was P. Owezarski.

C. Ge, Z. Sun, and N. Wang are with the Centre for Communication Systems Research, University of Surrey, Guildford GU2 7XH, U.K. (e-mail: C.Ge@surrey.ac.uk; Z.Sun@surrey.ac.uk; N.Wang@surrey.ac.uk).

K. Xu is with Tsinghua University, Beijing 100084, China (e-mail: xuke@tsinghua.edu.cn).

J. Wu is with Bell Laboratories, Shanghai 201206, China (e-mail: wujs@ieee.org).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNSM.2014.2346956

resolving requests to the fewest DCs and shutting down as many DCs as possible, the CDN operator can assure better QoS through e.g., prohibiting inter-domain request resolution. Although energy saving is traded off in this way, the energy-QoS trade-off is balanced better, which leads to both effective energy saving and assured QoS performance.

Our contributions in this paper are as follows. Firstly, we establish a holistic optimization formulation for the energy minimization problem, whose objective is to jointly optimize servers' and cooling systems' energy consumption among DCs in a CDN. To the best of authors' knowledge, it is the first in the literature to perform such a joint optimization through both server sleeping and DC shutdown simultaneously.

Secondly, we develop a practical algorithm named Min-DC-LD for the formulated problem. Min-DC-LD is a greedy heuristic that resolves requests to as few DCs as possible, which creates opportunities for some DCs to become idle and get shut down (especially during off-peak hours). Meanwhile, it enforces QoS constraints to maintain a well-balanced energy-QoS trade-off. Firstly, it restricts all requests to be resolved within their local domains to avoid unnecessary inter-domain content traffic. Secondly, it allows the CDN operator to specify the maximum distance between a request's source node and its designated DC, which suits a variety of QoS requirements by web applications with different tolerances of end-to-end delay. Furthermore, we show our algorithm's practicality in modern CDN infrastructures through explaining its participation in the request resolution process.

Thirdly, we analytically derive a theoretical lower bound to the formulated optimization problem. Such a lower bound can serve as a benchmark for Min-DC-LD and other algorithm's energy-saving performance. Since a problem's optimal objective is bound to be between its lower bound and any polynomial-time algorithm's performance, if the gap between lower bound's and Min-DC-LD's result is sufficiently small, our algorithm's performance is guaranteed to be near-optimal.

We carried out simulations based on real cross-domain network topology and workload trace, and the results are as follows. 1) Min-DC-LD is capable of achieving an energy-saving gain of up to 62.1% over existing CDN energy-saving schemes [4], and such gain is guaranteed to be near-optimal by our derived lower bound. 2) While producing its maximum energy-saving gain, Min-DC-LD's end-to-end delay meets the requirements of typical real-time web applications. Moreover, better latency and server response time can be achieved through trade-off in energy-saving performance. 3) The gap between our derived lower bound and Min-DC-LD's results is always less than 24.8% with median values between 6.2% and 11.7%, which makes the lower bound a suitable benchmark for other polynomial-time algorithms for the problem.

II. Problem Formulation

Consider a CDN infrastructure that covers multiple ISP autonomous domains with n Point-of-Presence (PoP) nodes in total. PoP nodes are where end-users' requests are aggregated in ISP backbone networks. A set of DCs are deployed at locations close to a subset of PoP nodes, and each DC contains one or more server clusters to resolve user requests. The incoming request volume (averaged per second) at each PoP node j ($j = 1 \ldots n$) is denoted by r_j in request/s. Each request from PoP j is resolved to some DC i ($i = 1 \ldots m$) designated by the CDN's request mapping system, and the overall request volume resolved from any PoP j to any DC i is represented by x_{ij} (in request/s). For each DC i, its service capability is denoted by Y_i, which refers to the maximum number of requests it can handle concurrently when all of its servers are active. Furthermore, its utilization, u_i, is defined as

$$u_i \triangleq \frac{\sum_{j=1}^{n} x_{ij}}{Y_i} \qquad (i = 1, \ldots, m). \tag{1}$$

Since we consider the option of shutting down entire DCs, a binary variable δ_i is introduced to indicate the on/off status of each DC i. DC i has zero utilization if it is shut down (i.e., $\delta_i = 0$). If DC i is active (i.e., $\delta_i = 1$), u_i's value must be between 0 and 1 to avoid overloading DC i. δ_i and u_i's relationship is summarized as

$$\begin{cases} 0 \le u_i \le 1, & \text{if } \delta_i = 1 \\ u_i = 0, & \text{if } \delta_i = 0 \end{cases} \quad (i = 1, \ldots, m). \tag{2}$$

We view each DC i as a set of servers when considering its request resolution and power consumption. Firstly, request resolution are performed at PoP-to-DC level only, and we assume that requests resolved to a DC are automatically assigned internally to a suitable server through existing DC load balancing mechanism [11]. Furthermore, we assume that each DC automatically puts its idle servers to the sleep mode with respect to its utilization, and u_i indicates the proportion of servers that are active in DC i [4].

In practice, a modern CDN infrastructure contains two main types of DCs, i.e., back-end and surrogate DCs. There are typically a few back-end DCs and many geographically distributed surrogate DCs. The main reason for such setup is that content popularity generally follow Zipf distribution with a long Pareto tail [12]. Typically, more than half of content objects are requested by users infrequently, while a small subset of them are frequently requested [13]. Therefore, in practice, only the popular contents are cached at distributed surrogate DCs for reduced latency [14]. In contrast, the unpopular ones are stored at back-end DCs for infrequent access [15].

In this paper, we assume that each surrogate DC stores a full copy of all *popular* content objects [16], and all unpopular content objects are stored at *only* the back-end DC [15]. It can be inferred that back-end DCs can never be shut down under this circumstance, since they are the only available DCs to resolve requests for unpopular contents. Therefore, from energy saving's perspective, we do not include back-end DCs in our optimization problem. For the rest of this paper, we use the term "DC" to refer to surrogate DCs only.

Based on the assumptions above, we calculate power consumption of servers and cooling systems of each DC i (denoted by P_i^{svr} and P_i^{cool} respectively) as follows.

Firstly, P_i^{svr} is defined by

$$P_i^{\text{svr}} \triangleq P_{i,\text{svr}}^{\text{peak}} u_i + P_{i,\text{svr}}^{\text{slp}} (1 - u_i) \tag{3}$$

where $P_{i,\text{svr}}^{\text{slp}}$ and $P_{i,\text{svr}}^{\text{peak}}$ refer to *aggregated* power consumption of servers in DC i when all of them are sleeping or fully-loaded respectively.

Note that unlike other related work in the literature [4], [10], we do not explicitly model power consumption of individual servers in each DC. Instead, we model the entire DC's power consumption as a whole. This is due to the following reasons. Firstly, content servers in CDN DCs are typically homogeneous with identical capacities and power characteristics [10]. Secondly, based on our energy-saving strategy, the active servers in each DC will have very high utilization since the sleeping servers' loads have been migrated to them. Hence, they are expected to consume near-peak power consumption.[1] Due to the two reasons above, $P_{i,\text{svr}}^{\text{peak}} u_i$ and $P_{i,\text{svr}}^{\text{slp}}(1-u_i)$ can represent the power consumption of *active* and *sleeping* servers respectively at DC i with utilization u_i.

Secondly, P_i^{cool} is defined by

$$P_i^{\text{cool}} \triangleq P_{i,\text{cool}}^{\text{peak}} \left(A \cdot \delta_i + B \cdot u_i + C \cdot u_i^2\right) + P_{i,\text{cool}}^{\text{slp}}(1-\delta_i) \tag{4}$$

where $P_{i,\text{cool}}^{\text{peak}}$ and $P_{i,\text{cool}}^{\text{slp}}$ refers to the power consumption of cooling system in DC i when it is fully loaded or shut down respectively, and coefficients A, B, and C are regression parameters calculated through the functions in [9] given the DC outdoor and indoor dry bulb temperature.

In a typical modern DC, its cooling power consumption is dominated by its chiller plant's compressor [17], whose power consumption is a quadratic function of its utilization [9]. We estimate a DC's chiller utilization as follows. Firstly, the amount of heat that is removed by the cooling system roughly equals the amount of heat dissipated by servers in a DC (for simplicity, we do not consider the heat exchange caused by different DC layout). Secondly, although most DC operators tend to over-provision cooling system's capacity, we conservatively assume that a cooling system's maximum heat removal capacity matches the maximum overall heat dissipation of all servers in that DC. Hence, a DC's chiller utilization can be estimated to equal its server utilization u_i.

In each DC i, the following relationship between $P_{i,\text{cool}}^{\text{peak}}$ and $P_{i,\text{svr}}^{\text{peak}}$ holds:

$$\text{PUE} = \frac{P_{i,\text{svr}}^{\text{peak}} + P_{i,\text{cool}}^{\text{peak}}}{P_{i,\text{svr}}^{\text{peak}}} \tag{5}$$

where Power Usage Effectiveness (PUE) is an industry standard metric that indicates a DC's energy efficiency. A DC industry survey has reported average values of 1.8 (2012) and 1.67 (2013) [8], while another industrial source reported the figures of 2.8 (2012) and 2.9 (2013) in North America [18]. Note that since servers and cooling systems dominate a DC's power consumption, we do not consider other DC components in this paper [19].

We now present the formulation of the energy optimization problem:

[1]In practice, to avoid poor server response time caused by overloading, an utilization threshold limit can be applied to the active servers.

Problem (P): Given $\boldsymbol{r} = (r_j) \in \mathbb{Z}_+^n$ and $\boldsymbol{Y} = (Y_i) \in \mathbb{R}_+^m$, find $\boldsymbol{X} = (x_{ij}) \in \mathbb{Z}_{\geq 0}^{m \times n}$ and $\boldsymbol{\delta} = (\delta_i) \in \mathbb{Z}^m$ such that

$$\min_{x_{ij}, \delta_i} \sum_{i=1}^{m} \left(P_i^{\text{svr}} + P_i^{\text{cool}}\right) \tag{6}$$

subject to:

$$\sum_{i=1}^{m} x_{ij} = r_j \quad (j = 1, \ldots, n) \tag{7}$$

$$u_i \leq \delta_i \quad (i = 1, \ldots, m) \tag{8}$$

$$\delta_i \in \{0, 1\} \quad (i = 1, \ldots, m). \tag{9}$$

Equation (6) is the objective of minimizing overall server and cooling power consumption among all DCs. Constraint (7) guarantees 100% service availability through resolving all requests from each PoP node to one or more DCs. Constraints (8) and (9) enforce the relationship between δ_i and u_i as in (2).

Substituting (1), (3), and (4) into (6), the original objective function (6) is expressed as:

$$\min_{x_{ij}, \delta_i} \sum_{i=1}^{m} \left[a_i \cdot \delta_i + b_i \cdot \sum_{j=1}^{n} x_{ij} + c_i \cdot \left(\sum_{j=1}^{n} x_{ij} \right)^2 \right] \tag{10}$$

where

$$a_i = P_{i,\text{cool}}^{\text{peak}} \cdot A - P_{i,\text{cool}}^{\text{slp}}$$

$$b_i = \frac{1}{Y_i} \left(P_{i,\text{svr}}^{\text{peak}} - P_{i,\text{svr}}^{\text{slp}} + P_{i,\text{cool}}^{\text{peak}} \cdot B \right)$$

$$c_i = \frac{1}{Y_i^2} \cdot P_{i,\text{cool}}^{\text{peak}} \cdot C.$$

Similarly, we substitute (1) into constraint (8) to get:

$$\frac{1}{Y_i} \sum_{j=1}^{n} x_{ij} \leq \delta_i \quad (i = 1, \ldots, m) \tag{11}$$

which replaces constraint (8) in the original formulation.

Problem (P) has been shown to be NP-hard in [20] since it can be reduced from a smooth non-convex nonlinear programming problem. Therefore, we derive a lower bound to the problem's optimal objective.

III. Deriving a Lower Bound to (P)

In this section, we employ Lagrangian relaxation to derive a lower bound to (P). Among the constraints in (P), we choose to apply Lagrangian relaxation on constraint (11). The resulting problem after relaxation is:

Problem (LP): Given $\boldsymbol{r} = (r_j) \in \mathbb{Z}_+^n$ and $\boldsymbol{Y} = (Y_i) \in \mathbb{R}_+^m$, find $\boldsymbol{X} = (x_{ij}) \in \mathbb{Z}_{\geq 0}^{m \times n}$, $\boldsymbol{\delta} = (\delta_i) \in \mathbb{Z}^m$ and $\boldsymbol{\lambda} = (\lambda_i) \in \mathbb{R}_+^m$ such that

$$\min_{x_{ij}, \delta_i, \lambda_i} \sum_{i=1}^{m} \left[a_i \cdot \delta_i + b_i \cdot \sum_{j=1}^{n} x_{ij} + c_i \cdot \left(\sum_{j=1}^{n} x_{ij} \right)^2 \right] + \sum_{i=1}^{m} \lambda_i \left(\frac{1}{Y_i} \sum_{j=1}^{n} x_{ij} - \delta_i \right) \tag{12}$$

subject to:

$$\sum_{i=1}^{m} x_{ij} = r_j \qquad (j = 1, \ldots, n) \tag{13}$$

$$\delta_i \in \{0, 1\} \qquad (i = 1, \ldots, m) \tag{14}$$

where $\boldsymbol{\lambda}$ is a set of m Lagrangian multipliers.

For notational convenience, we define ϕ as below:

Definition 1: Given any optimization problem (OP), the value of its optimal objective is defined as $\phi(\text{OP})$.

It can be inferred from constraint (11) that since $(\lambda_i) \in \mathbb{R}_+^m$, $\phi(\text{LP})$ is lower bound to $\phi(\text{P})$.

Following the relaxation, it is further observed that (LP) is the sum of the following two sub-problems:

Problem (LP1):

$$\min_{\delta_i, \lambda_i} \sum_{i=1}^{m} (a_i - \lambda_i)\delta_i \tag{15}$$

subject to:

$$\delta_i \in \{0, 1\} \quad (i = 1, \ldots, m). \tag{16}$$

Problem (LP2):

$$\min_{x_{ij}, \lambda_i} \sum_{i=1}^{m} \sum_{j=1}^{n} \left(b_i + \frac{\lambda_i}{Y_i} \right) x_{ij} + \sum_{i=1}^{m} c_i \left(\sum_{j=1}^{n} x_{ij} \right)^2 \tag{17}$$

subject to:

$$\sum_{i=1}^{m} x_{ij} = r_j \qquad (j = 1, \ldots, n). \tag{18}$$

Firstly, (LP1) is a simple unconstrained 0–1 optimization problem that can be solved through

$$\delta_i = \begin{cases} 1 & \text{if } a_i < \lambda_i \\ 0 & \text{otherwise} \end{cases} \quad (i = 1, \ldots, m). \tag{19}$$

Secondly, (LP2) is a quadratic knapsack problem which is NP-hard as it can reduced from the clique problem [21]. Hence, instead of solving $\phi(\text{LP2})$ directly, we try to find a lower bound to $\phi(\text{LP2})$ (denoted by LB(LP2)), since $\phi(\text{LP1}) + \text{LB(LP2)}$ is also a lower bound to $\phi(\text{P})$. To find LB(LP2), consider the problem with the objective below:

Problem (LB-LP2):

$$\min_{x_{ij}, \lambda_i} \sum_{i=1}^{m} \sum_{j=1}^{n} \left(b_i + \frac{\lambda_i}{Y_i} \right) x_{ij} + \sum_{i=1}^{m} \sum_{j=1}^{n} c_i x_{ij}^2 \tag{20}$$

subject to (18).

We claim that $\phi(\text{LB} - \text{LP2})$ is a valid lower bound to $\phi(\text{LP2})$. Note the difference between (20) and (17), which equals $\sum_{i=1}^{m} \sum_{j=1}^{n} c_i x_{ij}^2 - \sum_{i=1}^{m} c_i (\sum_{j=1}^{n} x_{ij})^2$, is always negative since $(x_{ij}) \in \mathbb{Z}_{\geq 0}$. Hence, $\phi(\text{LB} - \text{LP2}) < \phi(\text{LP2})$ holds.

A. Solving ϕ(LB-LP2) in Polynomial Time

Problem (LB-LP2) is in the form of a special "separable" quadratic convex problem with a fixed number of linear constraints, whose optimal objective can be found in polynomial time [22]. The process of solving $\phi(\text{LB-LP2})$ is as follows.

Firstly, (LB-LP2) can be decomposed into the sum of n optimization problems, and each sub-problem has its objective as the nth part of the decomposed (21). Each sub-problem, denoted by (LB-LP2_j) $(j = 1, \ldots, n)$, is described as below:

Problem $(LB\text{-}LP2_j)$*:*

$$\min_{x_{ij}, \lambda_i} \sum_{i=1}^{m} \left(b_i + \frac{\lambda_i}{Y_i} \right) x_{ij} + \sum_{i=1}^{m} c_i x_{ij}^2 \tag{21}$$

subject to:

$$\sum_{i=1}^{m} x_{ij} = r_j. \tag{22}$$

For each (LB-LP2_j) and any fixed $\boldsymbol{\lambda}$, we have:

Lemma 1: With j fixed, vector $\boldsymbol{X_j} = (x_{ij}) \in \mathbb{Z}_{\geq 0}^m$ that satisfies (22) is an optimal solution to (LB-LP2_j) if and only if $\exists \mu_j \in \mathbb{R}$ such that

$$x_{ij} = \begin{cases} \frac{\mu_j - (b_i + \lambda_i / Y_i)}{c_i} & \text{if } b_i + \frac{\lambda_i}{Y_i} < \mu_j \\ 0 & \text{if } b_i + \frac{\lambda_i}{Y_i} \geq \mu_j \end{cases} \quad (i = 1, \ldots, m) \tag{23}$$

Proof: Proof of Lemma 1 is given in [22], [23]. ■

It takes $O(m)$ time to search through the set of $(b_i + \lambda_i / Y_i | i = 1, \ldots, m)$ to identify the indices of i such that $b_i + \lambda_i / Y_i < \mu_j$. Denoting such set of i as I, we get from (21) and (22) that

$$\sum_{i \in I} \frac{\mu_j - (b_i + \lambda_i / Y_i)}{c_i} = r_j \tag{24}$$

Denote the optimal solution to (LB-LP2_j) as $\boldsymbol{X_j^*} = (x_{ij}^*)$. Substituting (24) into (23), we get that for each (LB-LP2_j):

$$x_{ij}^* = \frac{\frac{r_j + \sum_{i \in I} \frac{b_i + \lambda_i / Y_i}{c_i}}{\sum_{i \in I} 1 / Y_i} - (b_i + \lambda_i / Y_i)}{c_i} \tag{25}$$

Note that (x_{ij}^*) can be represented as a function of solely $\boldsymbol{\lambda}$. Therefore, if we can identify the set $\boldsymbol{\lambda}$ that produces (x_{ij}^*), then this set, denoted by $\boldsymbol{\lambda}^*$, produces $\phi(\text{LB-LP2})$ as well through (25). It is known that $\phi(\boldsymbol{\lambda}^*)$ is concave, and maximizing $\phi(\boldsymbol{\lambda}^*)$ is equivalent to solving $\phi(\text{LB-LP2})$ [24].

Recall that $\phi(\boldsymbol{\lambda}^*) = \sum_{j=1}^{n} \phi_j(\boldsymbol{\lambda}^*)$ where $\phi_j(\boldsymbol{\lambda}^*)$ (ϕ_j in short) is the optimal objective value of (LB-LP2_j). Hence, the next step is to identify the set $\boldsymbol{\lambda}^*$ for each ϕ_j function. Consider each ϕ_j to have p_j hyperplane equations. For each ϕ_j, we compute its p_j functions and all the respective cells through computing intersections of components of ϕ_j. Altogether, we have at most $\sum_{j=1}^{n} p_j = O(n)$ equations and $\sum_{j=1}^{n} O(p_j^m) = O(n)$ cells of quadracity of all ϕ_j functions.

We firstly introduce procedure Multi-Dim-Search, which is used in identifying the cell containing $\boldsymbol{\lambda}^*$ in each ϕ_j via

multidimensional search [25]. We assume there exists an oracle which takes as input any hyperplane equation in $\mathbb{R}^m$, and outputs the position of $\boldsymbol{\lambda}^*$ with respect to this equation:

Procedure Multi-Dim-Search (k, γ)

Input: k hyperplane equations in $\mathbb{R}^m$ constant γ $(0 < \gamma < 1)$
Output: position of $\boldsymbol{\lambda}^*$ with respect to at least γk out of the given k equations
1: **begin**
2: **repeat**
3: after calling the oracle **once**: position of $\boldsymbol{\lambda}^*$ with respect to at least **half** of the hyperplanes in K can be identified
4: **remove** the hyperplanes whose position to $\boldsymbol{\lambda}^*$ have been identified
5: **until** position of $\boldsymbol{\lambda}^*$ with respect to at least γk out of given k hyperplanes have been identified
6: **end**

The main idea of Multi-Dim-Search is to repeatedly identify the location of $\boldsymbol{\lambda}^*$ with respect to a certain proportion of the given k hyperplanes that have not been visited in previous iterations. Within each iteration, the oracle is called once, and the position of $\boldsymbol{\lambda}^*$ with respect to at least half of the hyperplanes unvisited in previous iterations can be identified. The procedure terminates when a given threshold proportion (γ) of k hyperplanes' locations to $\boldsymbol{\lambda}^*$ has been identified.

Through Multi-Dim-Search, we develop a multi-stage algorithm to identify the cells containing $\boldsymbol{\lambda}^*$ within all ϕ_j's.

Algorithm 1 Identifying cells containing $\boldsymbol{\lambda}^*$ in all ϕ_j's

Input: $\boldsymbol{\phi} = (\phi_j)$: set of ϕ_j functions $(j = 1, \ldots, r_s)$ each ϕ_j function has p_j hyperplane equations
Output: cells in each ϕ_j that contain $\boldsymbol{\lambda}^*$
1: **begin**
2: $r_s \leftarrow n$ // Initialization: all ϕ_j's are put in $\boldsymbol{\phi}$
3: **repeat**
4: **begin stage** s
5: **for all** $\phi_j \in \boldsymbol{\phi}$ **do**
6: $\gamma_j \leftarrow 1 - 1/(2p_j)$
7: apply Multi-Dim-Search (p_j, γ_j) to ϕ_j
8: **end for**
9: for at least half of ϕ_j's in $\boldsymbol{\phi}$:
their cells containing $\boldsymbol{\lambda}^*$ are identified.
10: **remove** from $\boldsymbol{\phi}$: the ϕ_j's whose cells containing $\boldsymbol{\lambda}^*$ have been identified
11: $r_s \leftarrow |\boldsymbol{\phi}|$
12: $s \leftarrow s + 1$
13: **until** the cells containing $\boldsymbol{\lambda}^*$ are identified in all ϕ_j's
14: **end**

The key idea of the algorithm is to identify in each iteration the cells containing $\boldsymbol{\lambda}^*$ in at least half of the candidate ϕ_j's through intelligent setting of γ_j for each ϕ_j. Since the total number of p_j functions at stage 1 is $O(n)$, $O(\log n)$ stages are needed. Therefore, the algorithm can be finished in polynomial time due to Multi-Dim-Search's polynomial complexity.

After identifying the optimal $\boldsymbol{\lambda}^*$ for each problem (LB-LP2$_j$), its value can be substituted into (25) to calculate the optimal values of $\boldsymbol{X}_{\boldsymbol{j}}^* = (x_{ij}^*)$. Correspondingly, the set of (x_{ij}^*) can be used in (20) to calculate the optimal objective value of (LB-LP2). Now we have a valid lower bound, $\phi(\text{LP1}) + \phi(\text{LB-LP2})$, to the original problem (P).

IV. Heuristic Algorithm for (P)

In this section, we present a practical polynomial-time heuristic **Min-DC-LD** (where LD stands for local domain). It employs two energy-saving strategies: 1) putting idle servers to sleep within each DC; and 2) shutting down idle DCs through load unbalancing among DCs in the same CDN domain.

Min-DC-LD has two stages. The first stage performs localized request resolution and server sleeping within individual DCs, where all requests are resolved to their nearest available DC (subject to their load capabilities). Afterwards, based on the first stage's output, the second stage tries to redirect the initially mapped requests to alternative local-domain DCs while subjecting to QoS constraints, so that opportunities are created for some DCs to become idle and get shut down.

We firstly introduce procedure Map-to-Best-DC, which is called by both stages of Min-DC-LD.

Procedure Map-to-Best-DC$(j, r_j, \max_dist)$

Input: j: PoP node
r_j: request volume at j
max_dist: maximum allowed distance between a request's source PoP and its designated DC
Output: $(x_{ij}) \in \mathbb{Z}_{\geq 0}^m$: request volume mapped from PoP j to each DC i
1: **begin**
2: **repeat** through each DC i in PoP j's local domain
3: **identify** DC i that is unvisited and closest to j
4: **if** DC i is available **and** the distance between DC i and PoP $j < max\_dist$ **then**
5: map r_j to i subject to i's capacity, and denote mapped volume as x_{ij}
6: $r_j \leftarrow r_j - x_{ij}$
7: **end if**
8: mark DC i as visited
9: **until** $r_j == 0$
10: **end**

Basically, procedure Map-to-Best-DC is a greedy heuristic that resolves r_j requests at given PoP node j to DCs while subjecting to the following criteria. Firstly, the designated DC

must be within local domain of PoP j, and its distance to j must be smaller than $\max\_dist$. Secondly, the designated DC's request-handling capability must not be violated. These criteria are honored to assure service availability and end-to-end delay during DC shutdown operations, which effectively avoids deteriorated QoS and unnecessary inter-domain content traffic. The procedure terminates when all r_j requests have been resolved by one or more DCs, which ensures 100% service availability. Its runs in $O(m)$ time complexity.

The reason for us to explicitly specify $\max\_dist$ is that within a single CDN domain, the end-to-end delay between two nodes is approximately linear to their physical distance in km [26]. Specifically, an intra-domain distance of 300 and 500 km would lead to an end-to-end delay of around 10–15 and 20 ms respectively [10]. Different web applications have various requirements for latency. For example, most latency-sensitive applications require a latency of up to 30 ms [10], while a delay of up to 125–150 ms is specified for typical real-time web services [27]. These two values correspond to $\max\_dist$ of about 800 km and 3000 km respectively. Hence, the CDN operator will be able to specify different $\max\_dist$ values in our algorithm according to their specific latency requirements.

The algorithm Min-DC-LD is presented below.

Algorithm 2 Min-DC-LD

Input: set of PoP nodes j, $j = 1, \ldots, n$
$\boldsymbol{r} = (r_j) \in \mathbb{R}_+^n$: set of request volume at PoP j
Output: $\boldsymbol{X} = (x_{ij}) \in \mathbb{Z}_{\geq 0}^{m \times n}$: request volume mapped from each PoP j to each DC i

1: **begin**
// Stage 1
2: **for all** PoP node j ($j = 1, \ldots, n$) **do**
3: **call** Map-to-Best-DC($j, r_j, \max\_dist$)
4: **end for**
// Stage 2
5: **repeat** through each DC i ($i = 1, \ldots, m$)
6: **identify** DC i that is lowest-utilized and unvisited
7: **if** all requests currently mapped to DC i can be redirected to alternative DCs **then**
8: **redirect** requests and shut down DC i
9: **else** keep DC i active
10: **end if**
11: mark DC i as visited
12: **until** all DCs have been visited
13: **end**

In the first stage, Min-DC-LD iterates through each PoP node j and resolve its request volume r_j to its nearest available DCs by calling procedure Map-To-Best-DC (j, r_j). This stage produces *localized* request resolution between each PoP and each DC, which runs in $O(mn)$ time (n iterations of calling Map-to-Best-DC that is $O(m)$). Then, in the second stage, the algorithm iterates through each DC i to check if all of its currently-served requests can be redirected to alternative DCs, which is checked through applying Map-to-Best-DC on each PoP's requests that are currently mapped to DC i. If yes, then DC i can be shut down after its load has been completely redirected to alternative DCs. It is worth noting that request redirection during this stage will not interrupt ongoing content delivery sessions, which we will cover in Section V. The second stage runs in $O(m^2 n)$ (m iterations at line 5, $O(m)$ for line 6 and $O(mn)$ for line 7).

Note that although we assumed each DC has homogeneous servers in Section II, Min-DC-LD does not require such an assumption. The reason is that Min-DC-LD performs request resolution at PoP-to-DC level only, which does not require specific knowledge on each server's characteristic. Also, Min-DC-LD requires future request volume at PoP nodes as inputs, which needs to be accurately predicted based on historical monitoring results in a CDN. In practice, CDN workload normally follows a regular pattern [4], [28], [29], which is commonly modeled with queuing theory [12], [15]. More simplified approaches (like moving average algorithm) were also employed in related work. In this paper, we do not assume any specific load prediction technique, and consider that request volume is already predicted and provided as input. However, we plan to investigate and enhance Min-DC-LD's performance under irregular traffic pattern, e.g., unexpected bursts in traffic volume in our future work.

V. Practicality Considerations

In this section, we discuss some key practicality issues of our energy-aware scheme in modern CDN infrastructures.

A. Energy-Aware Request Resolution and Redirection

In Section I, we mentioned that strategic planning of user-to-DC request resolution is the key in balancing the energy-QoS trade-off. In practice, a CDN infrastructure is managed in a centralized manner, and decisions on request resolution from end-users to DCs are made by the request mapping system (RMS), which is part of the CDN's central management platform. Traditionally, many factors are considered in the decision-making process to ensure that users experience desired latency, which include network conditions, server utilization and predicted user demand at PoP nodes [11]. After the RMS makes the decision on request resolution, it issues corresponding instructions to DNS units that are distributed across ISP domains [11]. The above process is performed regularly, so that all DNS units can always receive up-to-date user-to-DC mapping information. With this information, when a user request reaches its local DNS unit, it can be directed to its designated DC for efficient content delivery.

To realize energy-aware request resolution, the RMS needs to take energy saving into account during the above decision-making process. Specifically, instead of following the traditional QoS-oriented approach, the RMS can execute the Min-DC-LD algorithm to determine the designated DCs of each end-user's content request. As described in Section IV, our algorithm is able to make request resolution decisions under the strategy of shutting down as many DCs as possible while respecting QoS constraints. Realizing energy awareness in the RMS also needs changes to be made to CDN infrastructure,

such as modifying the RMS's components to execute our algorithm and deploying controllers to DCs to perform shutdown and startup operations. Interested readers are referred to [30] for more details.

For each DC that is marked by the RMS eligible for shutdown, the following actions need to be done by the RMS. Firstly, it needs to ensure that no upcoming new request is mapped to the DC. Secondly, regarding the ongoing content delivery sessions on the DC, the operator has the options of either waiting for them to finish or redirecting them to alternative DCs. The RMS realizes these actions through issuing updated request resolution instructions (according to Min-DC-LD's output) to distributed DNS units in the CDN *beforehand*. Through these actions, a DC will have become fully idle before it is shut down, which avoids extra user-experienced delay incurred by DC on/off state transition.

Recall from Section II that besides DC shutdown, we also consider each DC to be able to automatically put its idle servers to the sleep mode. In this circumstance, some ongoing content delivery sessions may need to redirected to alternative servers within the same DC. Such redirection can be achieved seamlessly by modern DC's internal load balancing mechanism, which instructs alternative servers to serve content requests on behalf of the server that is going to sleep. This technique is widely adopted in modern DCs [11].

B. Server Sleeping and DC Shutdown

When saving CDN energy through server sleeping and DC shutdown, CDN operators often have the concerns over whether provisioning fewer active servers and DCs will deteriorate CDN performance. Specifically, the concerns are mainly over the overhead/complexity of powering on/off servers and DCs in terms of operational costs and the trade-off in QoS.

Firstly, putting idle servers in the sleep mode has become a common practice in energy efficient DCs, especially during off-peak hours. Modern servers with industry-standard energy efficiency can be put to sleep or powered on within several minutes [4]. However, there are a number of issues that need attention. Firstly, to cope with sudden increase in user demand, it is common for DC operators to keep a small pool of idle servers active in the DC. Furthermore, frequently turning on/off a server should be avoided since it can cause wear-and-tear effect on its hardware, which negatively affects its reliability. Since CDN workload normally follows a regular pattern, it is relatively easy to predict future user demand and accordingly plan the provisioning of active servers.

Secondly, although DC shutdown is a new strategy in CDN energy saving, it is not uncommon in the DC industry. For example, a DC operator can shut down a DC for maintenance purpose or during power outage. In practice, there is a checklist that needs to be carefully followed when starting up or shutting down a DC to avoid any service outage. Such a checklist is complicated and DC specific, since different DCs have different layout, hardware type and server virtualization level etc. Typically, it takes around 30 minutes or less to fully shut down or start up a DC [31]. This makes DC shutdown suitable for energy-saving purpose. However, frequent DC shutdown operations are not practically feasible due to the time duration it takes. Furthermore, too-often on/off state transitions will cause wear-and-tear on servers and other DC hardware components. Therefore, it is desired to limit the number of DC shutdown operations to e.g., maximum once per day.

Overall, both server sleeping and DC shutdown are practical energy-saving techniques in modern CDNs. Recall that normal CDN workload follows a regular pattern every 24 hours, which enables a daily off-peak "window" for DC shutdown. Therefore, as long as the frequency and duration of shutdown events are carefully orchestrated, and that future user request volume are accurately predicted, the overhead of server sleeping and DC shutdown operations can be limited to a low level.

VI. Performance Evaluation

In this section, we evaluate the energy-saving and QoS performances of **Min-DC-LD**. Two reference schemes are used. Firstly, **LB(P)** is used to assess Min-DC-LD's efficiency (i.e., close to optimal) in energy saving. Secondly, we use the scheme **Min-Dist** as a benchmark to examine our scheme's energy-saving gain and its trade-off in QoS performance. Min-Dist aims to save energy through server sleeping within DCs, but not via DC shutdown. While resolving requests in a localized manner for optimized QoS, Min-Dist reflects the state-of-the-art in CDN energy saving [4]. To suit our needs of modeling DC servers' and cooling system's energy consumption, we developed a standalone CDN simulator with Java and implemented the three schemes above.

A. Experimental Setup

1) CDN Topology: We use the real topologies from GEANT [32] and Internet2 [33] networks, which are two interconnected autonomous domains in Europe and U.S. respectively. Altogether, there are 34 PoP nodes that are distributed over Europe (25 nodes) and U.S. (9 nodes). We take each PoP's time zone into account—Europe covers the time zones from UTC to UTC+2, and U.S. covers UTC-8 to UTC-5. Overall, 9 DCs are deployed in the CDN (5 in GEANT and 4 in Internet2), which is typical considering geographical areas and population of the two domains [34]. According to the conventional CDN DC deployment strategy [14], these 9 DCs are deployed within proximity to the 9 PoP nodes whose associated cities have the highest local populations.

2) Workload Information: We use the web traffic trace from ClarkNet (an ISP covering Washington DC metro area) [35], which is very similar to other traces that appeared in recent publications [4], [5], [10]. The trace covers 7 consecutive days and contains all content requests that originated from Washington DC area during the period. Recall from Section II that we only optimize energy consumption of surrogate DCs, which contain copies of popular content objects only. In the ClarkNet trace, 49.7% of the content objects are considered as popular ones (having been requested by more than once), which contributed to 99.1% of the total request volume. Request volumes are averaged over disjoint 1-hour periods.

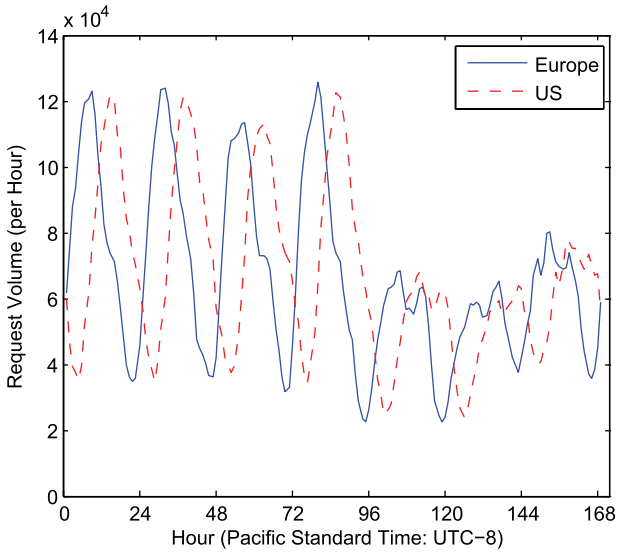


Fig. 1. 7-day content traffic trace, derived from ClarkNet WWW server trace [35].

In our experiments, the ClarkNet trace is approximated and applied to all PoP nodes according to the following approach. Firstly, we assume that end-users at all PoP nodes initiate content requests by following the same *pattern*. Such an assumption is based on observations in the literature [4], [36]. Afterwards, the number of requests at *each* PoP node is approximated by applying a PoP-specific scaling factor to the request volume in the trace, which is proportional to each PoP's local population. This is necessary since the population associated to each PoP, which affects its request volume, varies substantially in reality [37].

The resulted traffic trace is shown in Fig. 1. Note that we approximate user request volume in Europe and U.S. separately, where a time zone difference of 5 to 10 hours is present between them. As a result, their peak and off-peak hours are experienced at different time everyday, which needs to be taken into account during the joint energy optimization.

3) Experimental Scenarios: To reflect realistic CDN operational environments, we consider the following scenarios:

- **Over-provisioning ratio** refers to the ratio of total *provisioned* server capacity over *required* capacity to handle user requests at peak hours. The ratios of 1 : 1 and 2 : 1 are considered in our experiments. A ratio of 1 : 1 means that DCs in each domain have the service capacities that exactly match peak user demand in that domain, while a ratio of 2 : 1 means that half of the servers in each domain are needed to handle its peak user demand.
- **PUE**: Considering the variation in PUE values in the global DC industry [8], [18], we use the PUEs of 1.5, 2, and 3 in our experiments. The scenario with PUE of 1.5 describes some exceptionally energy efficient DCs, and the PUEs of 2 and 3 should be able to reflect typical DCs that are currently hosting CDN servers.
- **Outdoor air temperature**: we consider the temperatures of 50 °F (10 °C) and 85 °F (29.4 °C), which represent typical air temperatures in cold and hot areas respectively. Through these settings, we will evaluate air temperature's effect on our schemes' energy-saving performance.

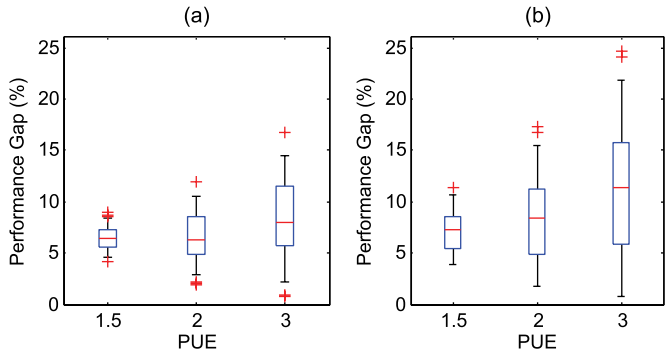


Fig. 2. Energy-saving performance gap between LB(P) and Min-DC-LD (85 °F). (a) 1 : 1 Over-Provisioning. (b) 2 : 1 Over-Provisioning.

4) Parameters: The parameters' values in the problem formulation are specified as follows.

In (1), service capability Y_i of each DC i is determined as below. Firstly, we calculate the peak aggregated request volume within *each* domain. Secondly, different over-provisioning ratios are applied to calculate the total *provisioned* server capability in each domain, which is then evenly distributed among its DCs. Under 1 : 1 over-provisioning ratio, each DC in Europe and US has the capability of 25939 and 32727 request/s respectively.

In (3), $P^{\text{peak}}_{i,\text{svr}}$ and $P^{\text{slp}}_{i,\text{svr}}$ for each DC i are respectively set to be 92 Watts and 5 Watts multiplied by the number of servers in DC i [4]. Each DC i's number of servers is calculated by dividing its service capability by 12 request/s (typical capacity of modern servers [12], [15]). Under 1 : 1 over-provisioning ratio, there are 2162 and 2728 servers within each DC in Europe and US respectively. Such a figure matches typical CDN server deployment scenario nowadays [4].

In (4), A, B, and C are determined by regression equations in [9] based on DC's indoor and outdoor dry bulb temperature. Note that we do not consider different indoor temperature, since it is normally set by e.g., ASHRAE recommendations [38]. $P^{\text{peak}}_{i,\text{cool}}$ is calculated with respect to $P^{\text{peak}}_{i,\text{svr}}$ through PUE based on (5).

B. Energy-Saving Performance

In this subsection, we evaluate the energy-saving performances of LB(P) and Min-DC-LD, which are compared against the reference scheme Min-Dist. We first evaluate Min-DC-LD's efficiency as a heuristic through assessing the gap between LB(P)'s and its results. As plotted in Fig. 2 with 99.7-percentile results, the gaps between the two schemes are shown to be small in all scenarios. Specifically, their gap is always less than 14.3% and 21.7% under 1 : 1 and 2 : 1 over-provisioning settings respectively, while the median is up to 8.2% and 11.7% under the two settings. Since the optimal objective of (P) is bound to be between LB(P) and Min-DC-LD's results, the implications of these results are two-fold. Firstly, LB(P) is sufficiently close to its optimal objective value, which means it can be used as an effective benchmark to evaluate other algorithms' performances. Secondly, Min-DC-LD's energy-saving gain over

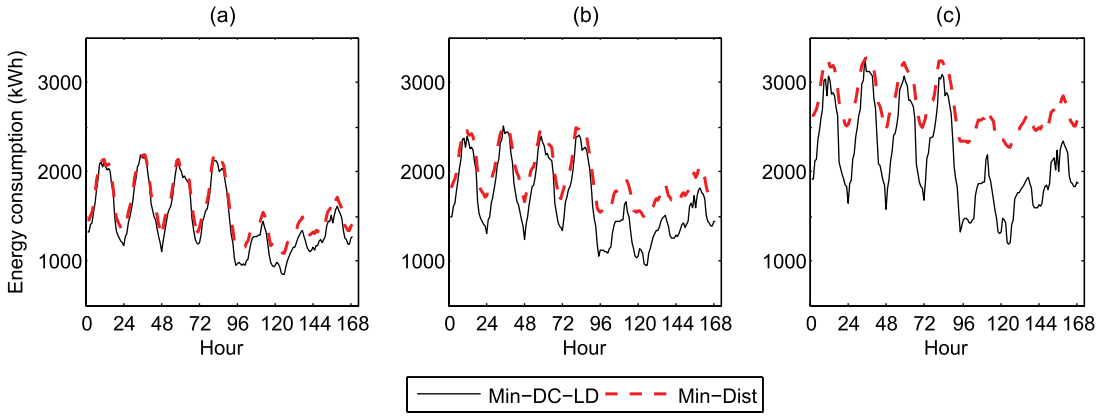


Fig. 3. Overall DC energy consumption per hour (1 : 1 over-provisioning, 85 °F). (a) PUE = 1.5. (b) PUE = 2. (c) PUE = 3.

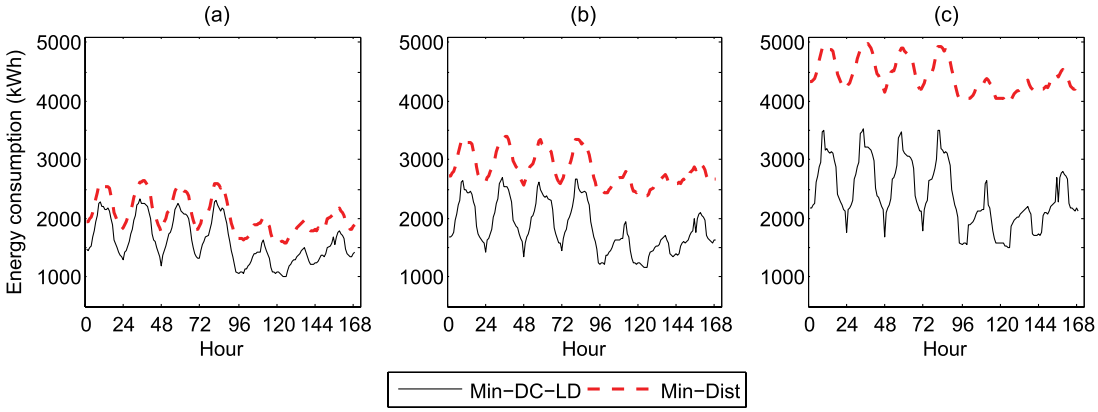


Fig. 4. Overall DC energy consumption per hour (2 : 1 over-provisioning, 85 °F). (a) PUE = 1.5. (b) PUE = 2. (c) PUE = 3.

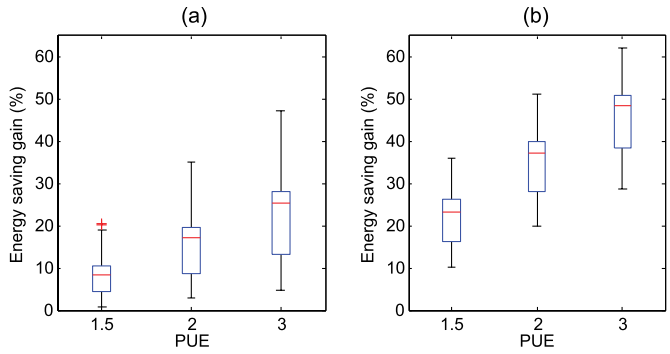


Fig. 5. Min-DC-LD's energy-saving gain over Min-Dist (85 °F). (a) 1 : 1 Over-Provisioning. (b) 2 : 1 Over-Provisioning.

Min-Dist is guaranteed to be near-optimal under all scenarios in our experiments.

Next, we evaluate Min-DC-LD's energy-saving gain over Min-Dist. The results of overall DC energy consumption per hour are shown in Figs. 3 and 4 respectively, and Min-DC-LD's 99.7-percentile energy-saving gain is plotted in Fig. 5. For now, we present the results under air temperature of 85 °F.

Firstly, under 1 : 1 over-provisioning ratio, Min-DC-LD was able to achieve an energy-saving gain of up to 47.1% over Min-Dist during off-peak hours. However, during peak hours, Min-DC-LD's advantage over Min-Dist was limited. The reason is that that Min-DC-LD's energy saving gain relies on shutting down the idle DCs whose loads are shifted to alternative DCs. However, during peak hours, *all* DCs were heavily utilized due to 1 : 1 over-provisioning. Hence, it was unlikely that any DC was able to become idle and get shut down, which eliminated Min-DC-LD's advantage over Min-Dist. This is within our expectation, because during a CDN's peak hours, it is of higher priority to guarantee QoS in e.g., service availability and end-to-end delay, which requires more active servers and DCs to be provisioned.

Secondly, under 2 : 1 over-provisioning ratio, Min-DC-LD was able to create more opportunities of DC shutdown due to more *spare* server resources provisioned in the CDN. As a result, it produced significant energy-saving gains over Min-Dist during both peak and off-peak hours. Specifically, the gains were 10.1% to 36% with PUE of 1.5, and 28.6% to 62.1% with PUE of 3. These results have shown Min-DC-LD's capability of saving DC energy through exploiting spare server resources and shutting down idle DCs.

A common observation under both scenarios above is that Min-DC-LD's achieved a higher energy-saving gain as PUE increased. Recall that the higher PUE is, the higher proportion that cooling system contributes to the overall DC energy consumption. Therefore, the DC shutdown technique is able to save more energy on cooling systems when PUE is higher (indicating poorer DC energy efficiency). Such an observation has important practical implications. For modern DCs with average PUEs of 2 to 3, the techniques of server sleeping and DC shutdown must be employed *together* to achieve

TABLE I
NUMBER OF DCS THAT IS SHUT DOWN BY MIN-DC-LD DURING THE TRACE (PUE $= 2, 85$ °F)

Number of Shut Down DCs (9 in total)	% of time during the Trace	
	1:1 Over-Provisioning	2:1 Over-Provisioning
9	-	-
8	-	-
7	-	13.3%
6	3.3%	45.7%
5	13.3%	35.7%
4	31.9%	5.2%
3	17.1%	-
2	22.9%	-
1	11%	-
0	0.5%	-

maximum energy saving. However, for DCs which have higher energy efficiency (with PUE of 1.5 or lower), the benefit of DC shutdown depends on how much spare server resource is provisioned in that CDN. If a conservative over-provisioning policy is employed, the extra energy saving that DC shutdown can achieve over server sleeping will be limited.

We further examine the number of DCs that Min-DC-LD is able to shut down with respect to different request volume during the trace, which is shown in Table I. Under $1:1$ over-provisioning setting, for 48.5% of the time during the 7-day trace, Min-DC-LD was able to shut down at least four DCs. Moreover, for 99.5% of the time, at least one DC was shut down. Under $2:1$ over-provisioning ratio, Min-DC-LD was able to shut down at least five DCs for 94.8% of the time. These results complement our earlier inference that given the same request volume, the more spare server resources that is provisioned in a CDN (i.e., higher over-provisioning ratio), the higher energy-saving gain that Min-DC-LD will be able to achieve. Note that PUE does not affect the results here, since DC shutdown decisions are determined by request resolution, which does not take cooling system into account.

So far, we have presented the results under outside air temperature of 85 °F. The results under 50 °F are very close to the ones under 85 °F with a less-than 1% gap. Therefore, we omit the results under 50 °F due to pagination limit.

C. QoS Performance

In this subsection, we evaluate the QoS performance of Min-DC-LD with respect to Min-Dist as a reference scheme. We use two metrics: 1) end-to-end delay experienced by content requests; and 2) server response time in DCs.

1) End-to-End Delay: Under both Min-DC-LD and Min-Dist, all requests are resolved within the local domain as their source PoP nodes. Therefore, a request's end-to-end delay can be approximated to be proportional to the network distance that it travels between its source PoP and its designated DC [26]. In this paper, we use IGP link weight to represent network distance since in GEANT and Internet2 networks, each link's weight is linearly proportional to its actual end-to-end delay [32], [33]. First, Min-Dist can achieve optimal end-to-end delay since it resolves all content requests in a localized manner [4]. Second, Min-DC-LD incurs extra network distance traveled by requests, as it redirects a subset of requests to alternative local-domain DCs to save more energy.

TABLE II
REPRESENTATIVE SNAPSHOT SCENARIOS

Scenario	Hour within the Trace	User Activity Level	
		Europe	*US*
#1	41	Off-Peak	Peak
#2	52	Peak	Off-Peak
#3	70	Off-Peak	Off-Peak
#4	105	Peak	Peak

To evaluate Min-DC-LD's trade-off in network distance, we picked four snapshots in the trace (shown in Table II) that are representative in user activity variation at each domain. For now, we do not limit the maximum network distance traveled by requests in Min-DC-LD. Within each scenario, we statistically analyzed and compared the network distances traveled by each request under Min-DC-LD and Min-Dist respectively. The 99.7-percentile results are shown in Figs. 6 and 7 under $1:1$ and $2:1$ over-provisioning settings respectively. Note that we ignore the outliers in Figs. 6 and 7 since they represent less than 0.001% of all requests.

It is observed from Figs. 6 and 7 that Min-DC-LD's network distance results were up to 2.4 and 8 times of Min-Dist's under $1:1$ and $2:1$ over-provisioning settings respectively. The gap between the two schemes' results became larger as more domains were in off-peak hours, since during off-peak hours, more idle servers exist and more opportunities for request concentration and DC shutdown are created. As a result, more request redirection takes place for load unbalancing, which leads to higher network distances traveled. Moreover, higher over-provisioning ratio also implies more idle servers, which explains the observations here.

We now quantitatively evaluate Min-DC-LD's end-to-end delay performance. Recall from earlier this section that a link's network distance (or weight) is linearly proportional to its latency. Based on [10], an intra-domain link with weight of 300 or 500 has the latency of 10–15 ms or 20 ms respectively. Hence, we constructed a linear relationship between a network path's distance and its end-to-end delay with these data, which was used to estimate the delay experienced by requests.

Under $1:1$ over-provisioning setting, the highest end-to-end delay under Min-DC-LD was 70 ms, which took place in scenario #3 (both off-peak hours). In scenarios #1 and #2, Min-Dist and Min-DC-LD produced delays of 25 ms and 45 ms respectively. Their delays were both around 50 ms in scenario #4. Under $2:1$ over-provisioning setting, Min-Dist's performance was improved due to more server resources available. In contrast, Min-DC-LD produced poorer end-to-end delay due to more request redirection, whose values were 125 ms, 67 ms, 160 ms and 60 ms in scenarios #1 to #4.

The results above have practical implications. Firstly, when at least one domain was not in off-peak hours, Min-DC-LD's delay performance met the need of typical real-time web applications (125–150 ms [27]). Secondly, when all domains were in off-peak hours (scenario #3), although aggressive energy saving through DC shutdown obtained near-optimal energy reduction,

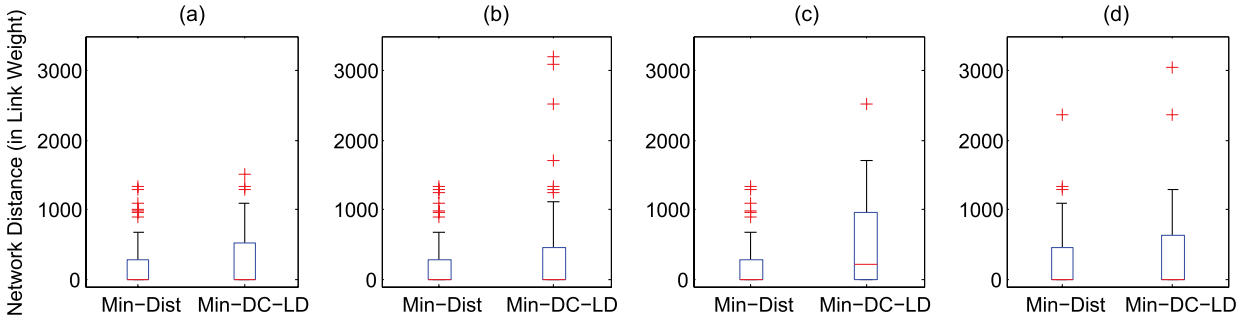


Fig. 6. Comparison on network distance traveled by content requests (1 : 1 over-provisioning, PUE = 2, 85 °F). (a) Snapshot #1. (b) Snapshot #2. (c) Snapshot #3. (d) Snapshot #4.

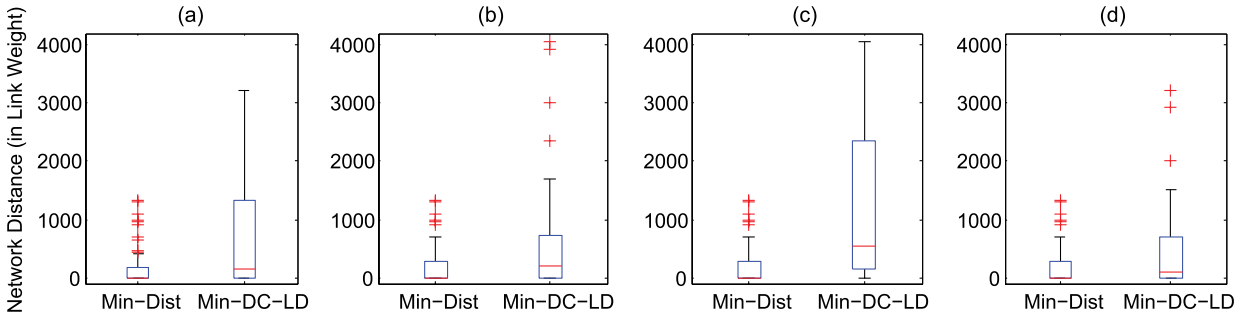


Fig. 7. Comparison on network distance traveled by content requests (2 : 1 over-provisioning, PUE = 2, 85 °F).(a) Snapshot #1. (b) Snapshot #2. (c) Snapshot #3. (d) Snapshot #4.

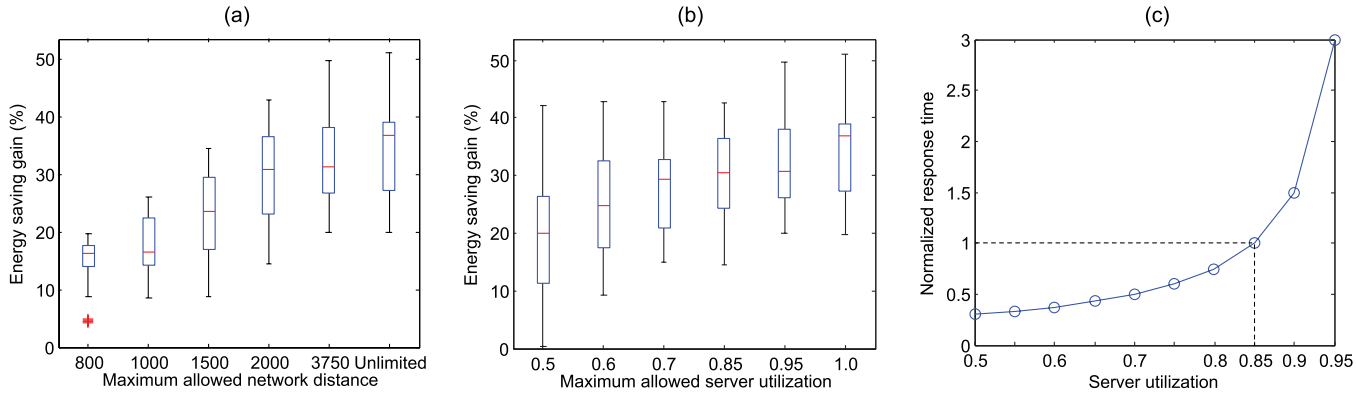


Fig. 8. Trade-off between Min-DC-LD's energy-saving gain and QoS performance, including end-to-end delay and server response time (2 : 1 over-provisioning, PUE = 2, 85 °F). (a) Trade-off: Max Net. Distance vs Energy Saving. (b) Trade-off: Server Util. vs Energy Saving. (c) Server Response Time.

the resulting QoS barely met 150 ms. Therefore, under these circumstances, energy saving will need to be compromised so that the CDN's delay performance can meet requirements by CDN-hosted web applications. To quantify such trade-off, we performed experiments on Min-DC-LD by specifying a range of $\max\_dist$ parameters and assessed the respective trade-off in its energy-saving gain. The results are shown in Fig. 8(a).

According to Fig. 8(a), to guarantee an end-to-end delay of 30 ms for latency-sensitive web applications (where $\max\_dist$ is set to 800), Min-DC-LD's maximum energy-saving gain was reduced from 51.1% to 20%. However, by limiting $\max\_dist$ to 3750 for an up-to-150 ms latency, Min-DC-LD was able to achieve near-optimal energy saving. The results in Fig. 8(a) provide practical guidelines on balancing energy-QoS trade-off with respect to latency requirements of different web applications.

2) Server Response Time: Typically, the server behavior in a DC can be modeled by an M/M/1 queue where request arrival follows Poisson distribution [12], [39]. Hence, the average server response time can be calculated by Little's Law of $1/\mu(1-\rho)$, where μ and ρ are the server's service rate and utilization respectively. Recall from our experimental setup that each server has a service rate of 12 req/s.

Under both Min-DC-LD and Min-Dist, to maximize server energy saving, each DC concentrates its load to the fewest active servers and put the remaining ones to sleep. Therefore, all servers will be utilized to the greatest extent that is allowed by the DC operator. On one hand, a server's response time is negatively correlated to its utilization. On the other hand, higher server utilization implies more energy saving gain, since DC loads can be handled by fewer servers. Hence, we investigate the trade-off between allowed server utilization and energy

TABLE III
REQUEST RESOLUTION DISTRIBUTION AMONG DATA CENTERS (2:1 OVER-PROVISIONING, PUE = 2, 85 °F)

Snapshot Scenario	Scheme	Total No. of Active DCs	Europe (with 5 DCs)			US (with 4 DCs)		
			No. of Active DCs	Average DC Load	Maximum DC Load	No. of Active DCs	Average DC Load	Maximum DC Load
#1 EU: Off-Peak US: Peak	LB(P)	3	3	90.7%	90.7%	0	-	-
	Min-DC-LD	3	1	93.2%	93.2%	2	90.9%	94.9%
	Min-Dist	9	5	18.6%	24.2%	4	35.5%	55.8%
#2 EU: Peak US: Off-Peak	LB(P)	3	2	92.3%	92.3%	1	92.5%	92.5%
	Min-DC-LD	4	3	74.3%	100%	1	62.2%	62.2%
	Min-Dist	9	5	44.6%	60.6%	4	15.6%	33.5%
#3 EU: Off-Peak US: Off-Peak	LB(P)	2	2	95.6%	95.6%	0	-	-
	Min-DC-LD	2	1	81.2%	81.2%	1	87.2%	87.2%
	Min-Dist	9	5	16.3%	22.3%	4	21.8%	33.6%
#4 EU: Peak US: Peak	LB(P)	4	4	97.6%	97.6%	0	-	-
	Min-DC-LD	4	2	96.0%	100%	2	78.5%	90.0%
	Min-Dist	9	5	38.4%	50.4%	4	39.3%	80.0%

saving gain in this part. Due to pagination limits, we only discuss the scenario of 2 : 1 over-provisioning and PUE = 2.

Firstly, through setting different values of maximum server utilization in our simulator, we obtain the 99.7-percentile results on energy saving gains as in Fig. 8(b). Note that setting maximum server utilization to 1.0 gives the upper bound on energy saving gain, which is hard to achieve in practice since it would cause poor response time. Secondly, we calculate server response time with a set of typical server utilization values between 0.5 and 0.95 [12], which is shown in Fig. 8(c). Note that we normalize the results with respect to the response time given by 85% server utilization. The reason is that we observed in the simulation that under 2 : 1 over-provisioning setup and Min-Dist scheme, the peak DC utilization is 60% in Europe and 85% in U.S. Hence, a server utilization of 85% leads to the worst-case (i.e., upper bound) response time *without* server sleeping in our simulation scenario.

It is directly observed from Fig. 8(b) and (c) that at 85% server utilization limit, the maximum energy saving gain is 8.5% less than the upper bound. The trade-off in median energy saving gain is 6.2%. If we set a higher server utilization limit (e.g., 95%), despite the response time becoming 3 times higher than under 85%, there would be hardly any increase in median energy saving gain. In contrast, if we set the server utilization limit to 70%, we can reduce the response time by half without sacrificing any energy saving performance. Furthermore, limiting server utilization to any point below 70% will not lead to significantly-better response time, but it will cost a considerable amount of energy saving.

D. Request Resolution Strategy

In this subsection, with respect to the four snapshot scenarios mentioned above, we examine more details on how content requests were resolved to DCs in each domain to realize the DC shutdown, which will help us better understand the trade-off between energy saving and QoS performance. The results under 2 : 1 over-provisioning setting are shown in Table III.

Firstly, it can be inferred that LB(P)'s strategy was to concentrate request resolution to as few DCs as possible. Moreover, it preferred mapping requests to DCs with *lower* load capability first (European DCs in this study). In scenarios #1, #3, and #4, no active DC was provisioned in US. Such strategies contributed to LB(P)'s advantage in energy-saving performance. However, its disadvantages are intuitive as well - requests from PoP nodes in US had be resolved to DCs in Europe, which would lead to deteriorated end-to-end QoS.

Min-DC-LD's total number of active DCs was similar to LB(P)'s. Within each domain, Min-DC-LD always provisioned as few number of DCs as possible with respect to its present request volume. In other words, when a domain is in off-peak hour, it will still have at least one active DC. Although such a strategy introduces gap between LB(P)'s and its energy-saving gain, it has important practical implications. Firstly, unlike LB(P)'s strategy, it enables all requests to be resolved within local domain, which is crucial in assuring QoS performance. Secondly, in case a spike in request volume occurs, it would be useful for a domain to have at least one active DC to absorb the load surge while more DCs are being powered up.

VII. RELATED WORK

Improving CDN content delivery efficiency is a traditional research topic, which involves strategic management of request resolution [12], [15], [34] and content cache placement [28], [40]. Since a few years ago, the topic of CDN's energy efficiency (especially on servers and DCs) started to emerge. Regarding saving energy in *individual* DCs, there has been a significant number of research work [3], [19], [39], [41], on which a comprehensive survey is available in [42]. In this section, we focus on the research that is related to saving energy costs among *multiple* DCs in a CDN.

So far, most work on CDN energy saving focused on the *server* part of DCs. Existing schemes can be generally divided into two categories based on their definitions of the term "energy cost" in Dollars or Joules/kWh. In both cases, energy cost reduction is achieved through strategic user-to-DC request resolution under different policies, so that utilization among multiple DCs are coordinated towards the objective of overall

energy reduction. Our work falls into the second category as we optimize energy in kWh.

In the first category, Qureshi *et al.* [26] proposed that CDN energy bill can be reduced through exploiting the diversity in electricity prices at different geographical areas. The idea was to resolve requests to DC sites where local electricity prices are cheaper, so that lower energy bill is incurred under the same amount of DC load. A similar approach as [26] was employed by Rao *et al.* in [43]. Later, a distributed version of the above strategy was developed by Xu *et al.* In [44], Liu *et al.* considered the possibility of using green renewable energy in some areas when performing request resolution. We did not compare our scheme against these work, since a cheaper electricity bill does not necessarily indicate less energy consumption and vice versa.

In the second category, Islam *et al.* [45] discussed the strategy of using less surrogate servers to resolve user requests in a CDN. Chiaraviglio *et al.* [46] developed a formulation that jointly optimizes energy consumption of CDN servers and ISP networking elements. However, such a scheme requires full information sharing between CDN operator and ISP, which is usually not realistic. Mathew *et al.* [4] developed offline and online algorithms to maximize CDN server energy reduction, which considered service availability and wear-and-tear on hardware caused by on/off state transitions. The algorithm in [4] was implemented in our experiments as the reference scheme Min-Dist. Gao *et al.* [5] developed exact and approximation algorithms to optimize the trade-off between CO_2 emission footprint and QoS in terms of latency. In these schemes, only servers' energy consumption was optimized in these schemes. In contrast, our scheme optimized both servers' and cooling systems' energy simultaneously. Furthermore, our work explicitly considered cross-domain CDNs, which made our study on energy-QoS trade-off more realistic as modern CDNs often covers multiple ISP domains.

Mathew *et al.* [10] studied the strategy of shutting down entire server clusters in DCs to save overall (server and cooling) energy costs, which stated that a cluster's cooling energy consumption can be eliminated when it is shut down. However, existing server sleeping schemes were not taken into account. In contrast, our work exploited both options of server sleeping and DC shutdown, which created more energy-saving opportunities. Furthermore, while only U.S. mainland was considered in [10], we studied cross-domain CDNs through considering time-zone difference and inter-domain traffic reduction, which had more practical implications.

VIII. Conclusion

In this paper, we investigated the joint optimization problem of cross-domain CDN's overall energy consumption, which included both server infrastructures and cooling systems among distributed DCs. We employed both strategies of server sleeping and DC shutdown, which were realized by our heuristic algorithm Min-DC-LD through concentrating request resolution to fewer DCs. QoS constraints were enforced in Min-DC-LD to ensure 100% service availability and that it meets end-to-end delay requirements. Through simulations under realistic scenarios, Min-DC-LD was able to achieve an energy-saving gain of up to 62.1% over Min-Dist (a server-sleeping-only scheme). These results were guaranteed to be near-optimal through our derived lower bound. We also studied the trade-off between energy saving gain and 1) end-to-end delay and 2) response time respectively, and showed that our scheme can balance the energy-QoS trade-off in a flexible manner. Considering our algorithm's practicality and low complexity, it is practical to integrate our scheme in modern CDN's request mapping system to realize energy-aware request resolution.

References

[1] H. Yin, X. Liu, G. Min, and C. Lin, "Content delivery networks: A bridge between emerging applications and future IP networks," *IEEE Netw.*, vol. 24, no. 4, pp. 52–56, Jul./Aug. 2010.

[2] J. Koomey. (2011). Growth in Data Center Electricity Use 2005 to 2010. [Online]. Available: http://www.analyticspress.com/datacenters.html

[3] G. Chen *et al.*, "Energy-aware server provisioning and load dispatching for connection-intensive internet services," in *Proc. USENIX NSDI*, Berkeley, CA, USA, 2008, pp. 337–350.

[4] V. Mathew, R. K. Sitaraman, and P. Shenoy, "Energy-aware load balancing in content delivery networks," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 954–962.

[5] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's not easy being green," in *Proc. SIGCOMM*, 2012, pp. 211–222.

[6] Y. Chen *et al.*, "Integrated management of application performance, power and cooling in data centers," in *Proc. IEEE NOMS*, Apr. 2010, pp. 615–622.

[7] R. Das, S. Yarlanki, H. Hamann, J. Kephart, and V. Lopez, "A unified approach to coordinated energy-management in data centers," in *Proc. CNSM*, Oct. 2011, pp. 1–5.

[8] M. Stansberry and J. Kudritzki, Uptime Institute 2012 data center industry survey, Uptime Inst., New York, NY, USA. [Online]. Available: http://www.uptimeinstitute.com/images/stories/Uptime_Institute_2012_Data_Industry_Survey.pdf

[9] *Nonresidential Alternative Calculation Method Approval Manual for the 2008 Building Energy Efficiency Standards*, California Energy Commission, Sacramento, CA, USA, Dec. 2008.

[10] V. Mathew, R. K. Sitaraman, and P. Shenoy, "Energy-efficient content delivery networks using cluster shutdown," in *Proc. 4th IGCC*, 2013, pp. 1–10.

[11] E. Nygren, R. K. Sitaraman, and J. Sun, "The Akamai network: A platform for high-performance internet applications," *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 3, pp. 2–19, Jul. 2010.

[12] T. Wu and D. Starobinski, "A comparative analysis of server selection in content replication networks," *IEEE/ACM Trans. Netw.*, vol. 16, no. 6, pp. 1461–1474, Dec. 2008.

[13] M. Busari and C. Williamson, "ProWGen: A synthetic workload generation tool for simulation evaluation of web proxy caches," *Comput. Netw.*, vol. 38, no. 6, pp. 779–794, Apr. 2002.

[14] R. Krishnan *et al.*, "Moving beyond end-to-end path information to optimize CDN performance," in *Proc. IMC*, 2009, pp. 190–201.

[15] S. Manfredi, F. Oliviero, and S. Romano, "A distributed control law for load balancing in content delivery networks," *IEEE/ACM Trans. Netw.*, vol. 21, no. 1, pp. 55–68, Feb. 2013.

[16] G. Rodolakis, S. Siachalou, and L. Georgiadis, "Replicated server placement with QoS constraints," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 10, pp. 1151–1162, Oct. 2006.

[17] S. Pelley, D. Meisner, T. F. Wenisch, and J. W. VanGlider, "Understanding and abstracting total data center power," in *Proc. WEED*, Jun. 2009, pp. 1–6.

[18] Digital Realty Trust, Colocation and Data Center Case Studies, San Francisco, CA, USA, Jan. 2013. [Online]. Available: http://www.digitalrealty.com/us/knowledge-center-us/?cat=Research

[19] Z. Abbasi, G. Varsamopoulos, and S. K. S. Gupta, "Tacoma: Server and workload management in internet data centers considering cooling-computing power trade-off and energy proportionality," *ACM Trans. Archit. Code Optim.*, vol. 9, no. 2, pp. 11:1–11:37, Jun. 2012.

[20] K. G. Murty and S. N. Kabadi, "Some NP-complete problems in quadratic and nonlinear programming," *Math. Programm.*, vol. 39, no. 2, pp. 117–129, Nov. 1987.

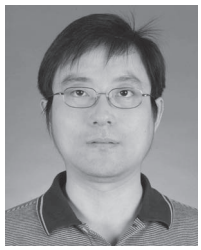
[21] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*. Berlin, Germany: Springer-Verlag, 2004.

[22] N. Megiddo and A. Tamir, "Linear time algorithms for some separable quadratic programming problems," *Oper. Res. Lett.*, vol. 13, no. 4, pp. 203–211, May 1993.
[23] T. Ibaraki and N. Katoh, *Resource Allocation Problems: Algorithmic Approaches*. Cambridge, MA, USA: MIT Press, 1988.
[24] Z.-Q. Luo and W. Yu, "An introduction to convex optimization for communications and signal processing," *IEEE J. Sel. Areas Commun.*, vol. 24, no. 8, pp. 1426–1438, Aug. 2006.
[25] N. Megiddo, "Linear programming in linear time when the dimension is fixed," *J. ACM*, vol. 31, no. 1, pp. 114–127, Jan. 1984.
[26] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 123–134, Oct. 2009.
[27] M. Chen, M. Ponec, S. Sengupta, J. Li, and P. A. Chou, "Utility maximization in peer-to-peer systems with applications to video conferencing," *IEEE/ACM Trans. Netw.*, vol. 20, no. 6, pp. 1681–1694, Dec. 2012.
[28] L. Qiu, V. Padmanabhan, and G. Voelker, "On the placement of web server replicas," in *Proc. IEEE INFOCOM*, 2001, vol. 3, pp. 1587–1596.
[29] P. Gill, M. Arlitt, Z. Li, and A. Mahanti, "Youtube traffic characterization: A view from the edge," in *Proc. ACM IMC*, 2007, pp. 15–28.
[30] C. Ge, N. Wang, and Z. Sun, "Energy-aware data center management in cross-domain content delivery networks," in *Proc. IEEE Online Conf. GreenCom*, Oct. 2013, pp. 88–95.
[31] *A Successful Data Center Migration*, Infosys, Bangalore, India, 2009.
[32] Geant Project Home. [Online]. Available: www.geant.net
[33] The Internet2 Network. [Online]. Available: www.internet2.edu/network/
[34] J. M. Almeida, D. L. Eager, M. K. Vernon, and S. J. Wright, "Minimizing delivery cost in scalable streaming content distribution systems," *IEEE Trans. Multimedia*, vol. 6, no. 2, pp. 356–365, Apr. 2004.
[35] Traces in the Internet Traffic Archive. [Online]. Available: http://ita.ee.lbl.gov/html/traces.html
[36] L. Cherkasova and M. Gupta, "Analysis of enterprise media server workloads: Access patterns, locality, content evolution, and rates of change," *IEEE/ACM Trans. Netw.*, vol. 12, no. 5, pp. 781–794, Oct. 2004.
[37] N. Kamiyama, T. Mori, R. Kawahara, S. Harada, and H. Hasegawa, "ISP-operated CDN," in *Proc. IEEE INFOCOM Workshops*, Rio de Janeiro, Brazil, 2009, pp. 49–54.
[38] *Thermal Guidelines for Data Processing Environments*, American Society Heating, Refrigerating Air-Conditioning Engineers, Inc., Atlanta, GA, USA, 2011.
[39] M. Lin, A. Wierman, L. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," *IEEE/ACM Trans. Netw.*, vol. 21, no. 5, pp. 1378–1391, Oct. 2013.
[40] T. Bektas, J.-F. Cordeau, E. Erkut, and G. Laporte, "Exact algorithms for the joint object placement and request routing problem in content distribution networks," *Comput. Oper. Res.*, vol. 35, no. 12, pp. 3860–3884, Dec. 2008.
[41] R. Urgaonkar, U. Kozat, K. Igarashi, and M. Neely, "Dynamic resource allocation and power management in virtualized data centers," in *Proc. IEEE NOMS*, 2010, pp. 479–486.
[42] C. Ge, Z. Sun, and N. Wang, "A survey of power-saving techniques on data centers and content delivery networks," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1334–1354, 2013.
[43] L. Rao, X. Liu, L. Xie, and W. Liu, "Minimizing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment," in *Proc. IEEE INFOCOM*, 2010, pp. 1–9.
[44] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Greening geographical load balancing," in *Proc. ACM SIGMETRICS*, 2011, pp. 233–244.
[45] S. ul Islam and J.-M. Pierson, "Evaluating energy consumption in CDN servers," in *ICT as Key Technology against Global Warming*. Berlin, Germany: Springer-Verlag, 2012, pp. 64–78.
[46] L. Chiaraviglio and I. Matta, "Greencoop: Cooperative green routing with energy-efficient servers," in *Proc. ACM e-Energy*, 2010, pp. 191–194.

Chang Ge (S'11) received the B.Eng. (Honours) degree in telecommunication engineering from Queen Mary, University of London, in 2009, and the M.Sc. degree in electronic engineering from the University of Surrey, U.K. in 2010. He is currently working toward the Ph.D. degree at the Centre for Communication Systems Research, University of Surrey. His research interests include energy efficiency in data centers and content delivery networks, as well as cache and energy management in next-generation Internet.

Zhili Sun (M'99) received the B.Sc. degree in mathematics from Nanjing University, Nanjing, China, in 1982 and the Ph.D. degree in computer science from Lancaster University, Lancaster, U.K., in 1991. He is currently the Chair of Communication Networking and has been with the Centre for Communication Systems Research, University of Surrey, Surrey, U.K., since 1993. He was a Postdoctoral Research Fellow with Queen Mary University of London, London, U.K., from 1989 to 1993. He has been a Principal Investigator and a Technical Cocoordinator on many projects within the EU Framework Programs, ESA, EPSRC, and industries. He has published more than 125 papers in international journals, book chapters, and conferences. He has published a book, as the sole author, entitled *Satellite Networking: Principles and Protocols* (Wiley, 2005), a book, as a contributing editor, entitled *IP Networking Over Next Generation Satellite Systems* (Springer, 2008), and another book, as a contributing editor to the fifth edition of the textbook, entitled *Satellite Communications Systems: Systems, Techniques and Technology* (Wiley, 2009). His research interests include wireless and sensor networks, satellite communications, mobile operating systems, traffic engineering, Internet protocols and architecture, quality of service, multicast, and security.

Ning Wang (M'12) received the B.Eng. (honours) degree from the Changchun University of Science and Technology, Changchun, China, in 1996, the M.Eng. degree from Nanyang University, Singapore, in 2000, and the Ph.D. degree from the University of Surrey, Surrey, U.K., in 2004, respectively. He is also a Senior Lecturer at the Centre for Communication Systems Research, University of Surrey, Surrey, U.K. His research interests mainly include energy-efficient networks, network resource management, information-centric networking and QoS mechanisms.



Ke Xu (SM'09) received the Ph.D. degree from the Department of Computer Science, Tsinghua University, Beijing, China, in 2001. Currently, he is a Full Professor with the Department of Computer Science, Tsinghua University. He has published more than 100 technical papers and holds 20 patents in the research areas of next generation Internet, P2P systems, Internet of Things (IoT), network virtualization and optimization. He is a member of ACM. He has guest edited several special issues in IEEE and Springer journals.

Jinsong Wu (SM'11) is the Founder and Founding Chair of the Technical Committee on Green Communications and Computing (TCGCC), IEEE Communications Society (established as Technical Subcommittee on Green Communications and Computing (TSCGCC) in 2011, elevated to TCGCC in 2013). He is an Associate Editor of *IEEE Communications Surveys & Tutorials, IEEE Systems Journal, and IEEE Access*, and the Founder and a Series Editor of the IEEE Series on Green Communications and Computing Networks for *IEEE Communications Magazine*. He has been a Guest Editor for *IEEE Communications Magazine*, *IEEE Systems Journal*, *IEEE Access*, IEEE TRANSACTIONS ON CLOUD COMPUTING, and *Elsevier Computer Networks*. He was the leading Editor of the comprehensive book *Green Communications: Theoretical Fundamentals, Algorithms, and Applications* (CRC, 2012).