IEEE/ACM Transactions on Networking

Greening Geographical Load Balancing

Journal:	*IEEE/ACM Transactions on Networking*
Manuscript ID:	TNET-2013-00132.R1
Manuscript Type:	Original Article
Date Submitted by the Author:	n/a
Complete List of Authors:	Liu, Zhenhua; California Institute of Technology, Computer Science Lin, Minghong; California Institute of Technology, Computing and Mathematical Sciences Wierman, Adam; California Institute of Technology, Computing and Mathematical Sciences Low, Steven; Caltech, CS & EE Andrew, Lachlan; Swinburne University of Technology, CAIA
Keywords:	data centers, geographical load balancing, renewable energy, distributed algorithms, demand response

Author response for "Greening Geographical Load Balancing"

We thank the reviewers for their detailed and thoughtful comments. We have incorporated the suggestions into the revision and believe that the paper is significantly improved.

In the following, we walk through the reviewers' comments and detail our thoughts and the changes we made to the paper as a result. Although we have adopted nearly all suggestions of the reviewers, one comment that we did not adopt is the comparison between our algorithms and state-of-the-art centralized solvers. This is because we believe there are indispensable advantages provided by distributed algorithms other than speedup. For example, distributed algorithms can increase scalability, deal with diversity in large systems more efficiently, reduce communication overhead, and be more robust to failure. Our distributed algorithms do not require centralizing information, and therefore can be deployed even when different parties (data centers, proxy routers) belong to different administration domains. For more details, please refer to our last two responses.

1. RESPONSES TO REVIEWS

1.1 Reviewer 1

Reviewer comment. I propose to accept this paper, for the following reasons:

- The paper tackles an interesting and current problem, of data-center load-balancing with both energy cost and green impact taken into account. While the problem might be solved centrally, the bulk of the paper considers the reasonable scenario where each proxy generating load on the data-centers must make its own decisions, leading to a distributed algorithm. With three distributed algorithms developed, the authors consider many aspects of their performance and convergence.

- The paper is well-rounded, supplying a strong and general model for the problem, theoretical proofs for optimization properties, and extensive experimental support.

- The paper is well written, very clear and with good structure. Assumptions and proofs are well detailed, and modeling justification as well as gaps for future work are clear.

Response. Thanks for your support.

1.2 Reviewer 2

Reviewer comment. Your distributed algorithms solve GLB-Q, which depends on your assumptions that (a) lost revenue is a linear function of delay, and (b) servers are power-unproportional (that is, each active server consumes the same amount of energy regardless of the load). Can you give some intuitions for how inaccurate your distributed algorithms would become if these assumptions are violated? For example, what happens if lost revenue is a piece-wise linear function, representing several different regime: (i) slow growth of lost revenue up to some acceptable delay, (ii) much sharper growth of lost revenue, (iii) slow growth of lost revenue again (most clients have abandoned). For (b), architects are working to make servers more power-proportional, so it may be important to examine the sensitivity to your assumption.

Response. We thank the reviewer for this helpful comment. Although in this paper we restrict our attention to GLB-Q in algorithm design, our distributed algorithms can be easily extended to more general cases. For example, any convex piece-wise linear lost revenue function containing, e.g., (i) slow growth of lost revenue up to some acceptable delay and (ii) much sharper growth of lost revenue. This will certainly make the expressions more complicated, but the theoretical component of the paper still works. The addition of (iii) slow growth of lost revenue again (most clients have abandoned) is tricky. One way to deal with it is to let the cost function go to infinity in this region because this region is what we want to avoid, then our algorithms apply naturally. Otherwise we need different techniques to deal with the problem since it makes the objective function non-convex, which is beyond the scope of this paper.

If servers are more power proportional, this can be captured by extending current model slightly. Assume the server power consumption at data center i follows the popular relationship

$$s_i = c_i + v_i \frac{\lambda_i}{m_i \mu_i},$$

where c_i and v_i are the constant and variable parts of server power consumption, respectively, and can be measured. μ_i is the service capacity of a single server and can be measured, therefore $\frac{\lambda_i}{m_i \mu_i}$ is the per server utilization level. Then the energy cost of data center i becomes

$$p_i m_i s_i = p_i \left(c_i m_i + \frac{v_i}{\mu_i} \lambda_i \right),$$

which is still jointly convex in m_i and λ_i. Therefore the theoretical results of our paper still apply and our algorithms can be adapted to this power form with minor changes.

Reviewer comment. II.B: While it is fine to focus on just the server costs of the data center, and it is true that "minimizing server energy consumption also reduces cooling and power distribution costs", you are ignoring an important cost factor in geographical load balancing: the different PUEs of different data centers due to geographical factors such as temperature (leading to higher cooling costs). In "Reducing Electricity Cost Through Virtual Machine Placement in High Performance Computing Clouds", Supercomputing 2011, Le et al. show that ignoring cooling costs in geographical load balancing can increase cost. You should mention that you are ignoring this factor.

Response. We agree that cooling cost is an important factor. We have added a discussion and references [16]-[18] in Section II.B and II.C of the revised version. Note that our model is general enough to include different cooling efficiencies of different data centers in the energy cost part. In particular, the problem of utilizing different cooling efficiencies has been studied in

Amip Shah et al. Thermal Management Considerations for Geographically Distributed Computing Infrastructures. ASME 2010.

For more details about cooling micro-grid modeling, please refer to Section 3.1 of our recent work

Zhenhua Liu et al. Renewable and cooling aware workload management for sustainable data centers. Sigmetrics 2012.

With added complexity, models from these can be incorporated.

Reviewer comment. Figure 3: why does delay increase with beta? If I understand your formulation correctly, r(d) represents lost revenue, so that larger beta means higher lost revenue given a delay. Thus, would not higher betas lead to lower delay while increasing energy cost?

Response. This is a typo. Thanks for pointing this out. We have corrected it to read "Circles, x-marks, and triangles correspond to $\beta = 0.4$, 1, and 2.5, respectively.".

Reviewer comment. II.B."Energy cost": You mention that a $g_i(t, m_i, y_i)$ that is strictly increasing in m_i and non-decreasing in y_i can represent a charging plan that includes a "demand charge" for the peak power usage period. I don't quite understand this if server power consumption is dependent on load. In particular, if server power consumption depends on load, a smaller y_i but with load concentrated in an interval (e.g., 15 minutes) can lead to higher peak power charge than a higher y_i but with load evenly spread out across t.

Response. We have clarified this point in Section II.B. $g_i(t, m_i, y_i)$ is the energy cost during timeslot t. Given $m_i(t)$ and $y_i(t)$ for all timeslots, the total power cost $\sum_t g_i(t, m_i, y_i)$ can capture the charging plan including a "demand charge" for the peak power usage period. We focus on the issue of peak pricing in our recent work "Data Center Demand Response: Avoiding the Coincident Peak via Workload Shifting and Local Generation". It requires slightly different approaches, but they can be merged. We also assume the length of each timeslot matches or is smaller than that used in the charging scheme.

Reviewer comment. - Equations (2) and (3a): I'm surprised that $D_i(t)$ is not multiplied by $|t|$ (I'm assuming that y_{ij} is a rate), since $g_i(t, m_i, y_i)$ gives the energy usage for all of t.

Response. The paper works in discrete time, with timeslots normalized to length 1. (Section II.A now explicitly says "We consider a discrete-time model with time step duration normalized to 1, such that...". Note that $g_i(t, m_i, y_i)$ gives the energy use only for a single time slot. The variable t identifies the time slot; it is not the duration of the time slot.

Reviewer comment. V.B.4: it seems counter-intuitive that poor estimates of d_{ij} and L_j have little effect on cost. I think it's worthwhile to expand a bit on this (even if space constraints require cutting a bit elsewhere).

Response. We have added more details on this point in Section V.B.4. Specifically, our sensitivity analysis shows a 20% prediction error (i.i.d. uniform distribution with mean absolute error 20%) in propagation delay d_{ij} only increases total cost by 2%. The reason behind this is that the total delay cost is comparable to the total energy cost, and network delay is roughly 30% of the total delay, therefore the network delay cost is roughly 15% of the total cost. Additionally, our solution is robust to moderate prediction error. For example, if the data center(s) chosen by a particular proxy are still optimal with prediction error, the prediction error does not change the optimality of our solution.

For load prediction error, the impact is minimal because of two reasons. For proxy, our solution provides how to spread the traffic among all data centers. When facing load prediction error, the proxy can still spread the traffic according to the fraction obtained by our solution and scale up/down according to prediction error. Usually this does not change the optimality of our solution. For each data center, it always chooses the optimal number of servers according to incoming load, even when there is load prediction error at proxies. This will greatly alleviate the negative impacts of load prediction error. Our results show a 20% prediction error (i.i.d. uniform distribution with mean absolute error 20%) in load L_j only increases total cost by less than 1%.

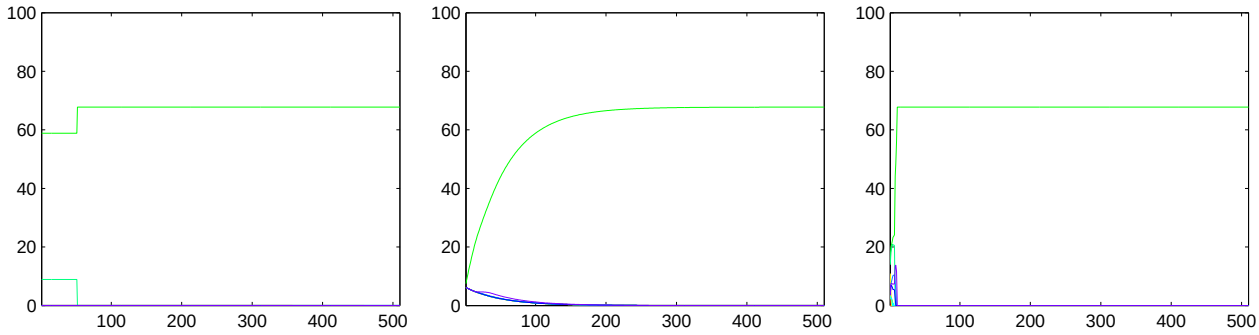
Reviewer comment. What beta did you use for Figure 4?

Response. We use $\beta = 1$ for figures in Section V except that in Figure 3 we vary β to study its impacts. We note this in V.A.3.

1.3 Reviewer 3

Reviewer comment. This paper studies how to explore geographical diversities of Internet systems to optimize performance, e.g., delay, energy cost as well as social impact such as consumption of green energies. This work proposes three distributed algorithms that provably converge to the optimal solutions. It also demonstrates the importance of dynamic pricing for achieving the benefit of brown energy reduction.

Overall the paper is well written with general problem formulations and rigorous proof. The third distributed al-



(a) Algorithm 1 (b) Algorithm 2 (c) Algorithm 3

Figure 1: Routing stability in our simulations.

gorithm has significant practicality for its sound convergence speed, in contrast to prior work that mostly applies Gauss-Seidel iterative algorithms.

One minor comment: in large-scale data centers, routing stability is an important concern. Have you compared shifts in routing table entries during the convergence for three algorithms respectively?

Response. The actual shifts in routing table entries for different algorithms depend delicately on parameter settings.

Intuitively, our second and third algorithms consider the interactions between different proxies, and therefore could reduce the routing oscillation. In particular, Algorithm 2 moves more conservatively, therefore probably has lowest oscillation.

In our simulations, routing is quite stable. Figure 1 in this response shows the traffic going from one proxy to all data centers for all our three algorithms over time.

We do not change the routing inside a data center.

1.4 Reviewer 4

Reviewer comment. With respect of the network, the modelling part consider only the delay among data centers. Delay is an input data taken from a present configuration. In this way, the delays that the new routing can produce is not taken into account. Furthermore no consideration about the energy consumption of the network is present.

Response. In this paper, we assume the data center's job requests are only a small fraction of the network traffic. Therefore the update in routing does not change network delay significantly. Also, our algorithms work at a timescale of minutes. At this timescale, it is reasonable to approximate the end-to-end delay by the average delay. Our algorithms react to the average delay at this timescale, not to the delay variations at second or ms timescale such like TCP does.

The energy consumption of the network is a big problem by itself and there are huge efforts on this topic. We did not aim to solve the problem in this paper. However, if the network power consumption is known then d_{ij} can simply be set to the delay plus the incremental energy cost of sending one request over the path from i to j. More general network power function can also be included if needed.

Our coauthor Steven Low also has recent work specifically on network power consumption

Lingwen Gan et al. Energy-efficient congestion control. Sigmetrics 2012.

Reviewer comment. The data center model is very simple, just considering the number of machine on (with continuos variables), with no consideration of switching on and off costs and consumptions.

Response. We definitely agree that we have made simplifying assumptions, however, we feel that this model and its associated results are a useful step towards results about a more detailed model. In this paper, we assume that servers are either active or in a low power state, and the switching cost is small as argued by the following paper.

David Meisner, et al. PowerNap: eliminating server idle power. ACM Sigplan Notices. Vol. 44(3). 2009.

We also see great efforts are undergone to improve the power proportionality of commercial servers, in that case our model applied naturally.

Switching costs are a more different issue to discuss. Note also our recent paper focused on the impacts of switching costs and showed with careful algorithm design, the impacts can be minimal.

Minghong Lin, et al. Online algorithms for geographical load balancing. IGCC 2012.

Reviewer comment. From the methodological point of view, the authors add a further algorithm for the resolution of the problem with respect of the conference paper. In my opinion there is no need of such an algorithm, or at least is not justified by the authors. First of all, the authors themselves claims that "Algorithm 1 [not distributed], converges, fast enough to provide near-optimal costs.".

Response. We agree with the reviewer that nowadays there is enough computing power in the cloud to solve many large problems centrally, e.g., Yahoo conducts on the order of a billion real-time options a day (though that's not exactly solved in a centralized manner). We emphasize, however, that distributed algorithms provide other (more) important advantages than potential speedup, and for some applications, these advantages are indispensable. For example:

1. In a large-scale system such as the one studied in the

paper, we can use local update instead of global update, and therefore react only when necessary. This is more efficient to deal with diversity in large *asynchronous* systems. In contrast, a naive centralized algorithm will need to execute periodically and to efficiently match the execution to the asynchronous changes over a large network, an appropriate update frequency needs to be chosen for the global problem (which is hard). Also the global problem needs to be broken down into subproblems so that only those subproblems whose states have changed needs to be solved, which probably will lead to something like the distributed algorithms developed in this paper.

2. Most importantly for deployment, since distributed algorithms do not require centralizing information, they can be deployed where different parties (data centers, proxy routers) belong to different administration domains (or even different third-part companies) and sharing of data among them is difficult. It also does not require customers to share their private information. In this situation, a centralized algorithm cannot be deployed.

3. Communication overhead: In our distributed algorithms, there is no need to transfer most information, e.g., network delay can be measured locally by each proxy. Detailed routing decisions for each proxy and each data center is not needed, but only the aggregated incoming load is measured by data center to decide the right number of active servers. This minimizes the communication overhead in the distributed system.

4. Our distributed solution is more robust to single-node failure compared with centralized solution. Even though we can use duplicated nodes for the centralized information collection and computation to improve, this can definitely bring more overhead for communication and synchronization.

We proposed three algorithms with different degrees of distributed computation. In Algorithm 1, proxies can only update one by one, which makes its convergence rate the slowest among all three algorithms. We therefore proposed Algorithm 2 in which proxies can update simultaneously, which improves the convergence rate, and increases responsiveness of all proxies. Due to the computational overhead of projection computation in Algorithm 2, we design Algorithm 3 with even better convergence while avoiding the need for projection in each iteration. Algorithm 3 is especially interesting for its low computational overhead and fast responsiveness.

From theoretical perspective, our algorithms provide guaranteed convergence even though the objective function does not satisfy smoothness condition, i.e., Lipschitz condition, which is required by traditional distributed algorithms. This can also be useful for the analysis of other distributed algorithms.

Reviewer comment. Second, models with linear constraints and a non linear objective function are nowadays solved by state of the art (commercial and free) solvers. For this reason, before developing a new algorithm is advisable to assess that these solvers are not able to solve the problem with enough accuracy or efficiency.

Response. As we just described, our focus is the design of distributed algorithms. Such algorithms are necessarily slower than centralized solutions. Thus the relevant comparisons are with other distributed algorithms and this is also our motivation for including all three algorithms, since this allows us to compare the convergence rates. Given the space constraints, we have chosen not to include the comparison with start-of-the-art centralized solvers.

Greening Geographical Load Balancing

Zhenhua Liu, Minghong Lin, Adam Wierman, Steven Low, *Fellow, IEEE* and
Lachlan L. H. Andrew, *Senior Member, IEEE*

***Abstract*—Energy expenditure has become a significant fraction of data center operating costs. Recently, "geographical load balancing" has been proposed to reduce energy cost by exploiting the electricity price differences across regions. However, this reduction of cost can paradoxically increase total energy use.**

We explore whether the geographical diversity of Internet-scale systems can also provide environmental gains. Specifically, we explore whether geographical load balancing can encourage use of "green" renewable energy and reduce use of "brown" fossil fuel energy. We make two contributions. First, we derive three distributed algorithms for achieving optimal geographical load balancing. Second, we show that if the price of electricity is proportional to the instantaneous fraction of the total energy that is brown, then geographical load balancing significantly reduces brown energy use. However, the benefits depend strongly on dynamic energy pricing and the form of pricing used.

***Index Terms*—data centers, geographical load balancing, renewable energy, distributed algorithms, demand response.**

I. Introduction

Increasingly, web services are provided by massive, geographically diverse "Internet-scale" distributed systems, some having several data centers each with hundreds of thousands of servers. Such data centers require many megawatts of electricity and so companies like Google and Microsoft pay tens of millions of dollars annually for electricity [1].

The enormous, and growing, energy demands of data centers have motivated research both in academia and industry on reducing energy usage, for both economic and environmental reasons. Engineering advances in cooling, virtualization, DC power, etc. have led to significant improvements in the Power Usage Effectiveness (PUE) of data centers; see [2]–[5]. Such work focuses on reducing the *energy use* of data centers and their components.

A different stream of research has focused on exploiting the geographical diversity of Internet-scale systems to reduce the *energy cost*. Specifically, a system with clusters at tens or hundreds of locations around the world can dynamically route requests/jobs to clusters based on proximity to the user, load, and local electricity price. Thus, dynamic geographical load balancing can balance the revenue lost due to increased delay against the electricity costs at each location.

The potential of geographical load balancing to provide significant cost savings for data centers is well known; see [1], [6]–[10] and the references therein. The goal of the current paper is different. Our goal is to explore the social impact of geographical load balancing systems. In particular, because GLB reduces the average price of electricity, it reduces the incentive to make other energy-saving tradeoffs.

In contrast to this negative consequence, geographical load balancing provides a huge opportunity for environmental benefit as the penetration of green, renewable energy sources increases. Specifically, an enormous challenge facing the electric grid is that of incorporating intermittent, non-dispatchable renewable sources such as wind and solar. Because generation supplied to the grid must be balanced by demand (i) instantaneously and (ii) locally (due to transmission losses and the prohibitive cost of high-capacity long-distance electricity transmission lines), renewable sources pose a significant challenge. A key technique for handling the non-dispatchability of renewable sources is *demand response*, which entails the grid adjusting the demand by changing the electricity price [11]. However, demand response entails a *local* customer curtailing use. In contrast, the demand of Internet-scale systems is flexible geographically; thus requests can be routed to different regions to "follow the renewables" to do the work in the right place, providing demand response without service interruption. Since data centers represent a significant and rapidly growing fraction of total electricity consumption, and the IT infrastructure with necessary knobs is already in place, geographical load balancing can provide an inexpensive approach for enabling large scale, global demand response.

The key to realizing the environmental benefits above is for data centers to move from the typical fixed price contracts that are now widely used toward some degree of dynamic pricing, with lower prices when renewable energy generation exceeds expectation. The current demand response markets provide a natural way for this transition to occur, and there is already evidence of some data centers participating in such markets [12].

The contribution of this paper is twofold. (1) We develop distributed algorithms for geographical load balancing with provable optimality guarantees. (2) We use the proposed algorithms to explore the feasibility and consequences of using geographical load balancing for demand response in the grid.

Contribution (1): To derive distributed geographical load balancing algorithms we use a simple but general model, described in detail in Section II. In it, each data center minimizes its cost, which is a linear combination of an energy cost and the lost revenue due to the delay of requests (which includes both network propagation delay and load-dependent queueing delay within a data center). The geographical load balancing algorithm must then dynamically decide both how requests should be routed to data centers and how to allocate capacity in each data center (e.g., speed scaling and how many servers are kept in active/energy-saving states).

In Section III, we characterize the optimal geographical load balancing solutions and show that they have practically appealing properties, such as sparse routing tables. In Section IV, we use the previous characterization to design three distributed algorithms which provably compute the optimal routing and provisioning decisions and require different degrees of coordination. The key challenge here is how to design distributed algorithms with guaranteed convergence without Lipschitz continuity. Finally, we evaluate the distributed algorithms using numeric simulation of a realistic, distributed, Internet-scale system (Section V). The results show that a cost saving of over 40% during light-traffic periods is possible.

This is an extension of our conference paper in SIGMETRICS 2011.
Zhenhua Liu, Minghong Lin, Adam Wierman, and Steven Low are with the Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA, 94706 USA e-mail: {zhenhua,mhlin,adamw,slow}@caltech.edu, Lachlan L.H. Andrew is with Swinburne University of Technology, Australia, e-mail: landrew@swin.edu.au.

Contribution (2): In Section VI we evaluate the feasibility and benefits of using geographical load balancing to facilitate the integration of renewable sources into the grid. We do this using a trace-driven numeric simulation of a realistic, distributed Internet-scale system in combination with real wind and solar energy generation traces over time.

When the data center incentive is aligned with the social objective for reducing brown energy by dynamically pricing electricity proportionally to the fraction of the total energy coming from brown sources, we show that "follow the renewables" routing ensues (see Figure 5), providing significant social benefit. We determine the wasted brown energy when prices are static, or are dynamic but do not align data center and social objectives enough, also later shown by [13].

II. Model and Notation

We now introduce the workload and data center models, followed by the geographical load balancing problem.

A. The workload model

We consider a discrete-time model with time step duration normalized to 1, such that routing and capacity provisioning decisions can be updated within a time slot. There is a (possibly long) interval of interest $t \in \{1, \ldots, T\}$. There are $|J|$ geographically concentrated sources of requests, i.e., "cities", and work consists of jobs that arrive at a mean arrival rate of $L_j(t)$ from source j at time t is. Jobs are assumed to be small, so that provisioning can be based on the $L_j(t)$. In practice, T could be a month and a timeslot length could be 1 hour. Our analytic results make no assumptions on $L_j(t)$; however numerical sections V and VI use measured traces to define $L_j(t)$.

B. The data center cost model

We model an Internet-scale system as a collection of $|N|$ geographically diverse data centers, where data center i is modeled as a collection of M_i homogeneous servers. The model focuses on two key control decisions of geographical load balancing at each time t: (i) determining $\lambda_{ij}(t)$, the amount of requests routed from source j to data center i; and (ii) determining $m_i(t) \in \{0, \ldots, M_i\}$, the number of active servers at data center i. Since Internet data centers typicall contain thousands of active servers, we neglect the integrality constraint on m_i. The system seeks to choose $\lambda_{ij}(t)$ and $m_i(t)$ in order to minimize cost during $[1, T]$. Depending on the system design these decisions may be centralized or decentralized. Section IV focuses on the algorithms for this.

Our model for data center costs focuses on the server costs of the data center.[1] We model costs by combining the *energy cost* and the *delay cost* (in terms of lost revenue). Note that, to simplify the model, we do not include the switching costs associated with cycling servers in and out of power-saving modes; however the approach of [14], [15] provides a natural way to incorporate such costs if desired.

Energy cost. To capture the geographical diversity and variation over time of energy costs, we let $g_i(t, m_i, \lambda_i)$ denote the energy cost for data center i during timeslot t given m_i active servers and arrival rate λ_i including cooling power [16]–[18]. . For every fixed t, we assume that $g_i(t, m_i, \lambda_i)$ is continuously differentiable in both m_i and λ_i, strictly increasing in m_i, non-decreasing in λ_i, and jointly convex in m_i and λ_i. This formulation is quite general. It can capture a wide range of models for power consumption, e.g., energy costs as an affine function of the load, see [19], or as a polynomial function of the speed, see [20], [21][2].

Defining $\lambda_i(t) = \sum_{j \in J} \lambda_{ij}(t), \forall t$, the total energy cost of data center i during timeslot t, denoted by $\mathcal{E}_i(t)$, is simply

$$\mathcal{E}_i(t) = g_i(t, m_i(t), \lambda_i(t)). \quad (1)$$

Delay cost. The delay cost captures the lost revenue incurred from the delay experienced by the requests. To model this, we define $r(d)$ as the lost revenue associated with average delay d. We assume that $r(d)$ is strictly increasing and convex in d.

We consider the two components of delay: the network delay while the request is outside the data center and the queueing delay within the data center.

Let $d_{ij}(t)$ denote the average *network delay* of requests from source j to data center i in timeslot t. Let $f_i(m_i, \lambda_i)$ rate of λ_i. We assume that f_i is strictly decreasing in m_i, strictly increasing in λ_i, and strictly convex in both m_i and λ_i. Further, for stability, we must have that $\lambda_i = 0$ or $\lambda_i < m_i \mu_i$, where μ_i is the service rate of a server at data center i. Thus, we define $f_i(m_i, \lambda_i) = \infty$ for $\lambda_i \geq m_i \mu_i$. For other m_i, we assume f_i is finite, continuous and differentiable. Note that these assumptions are satisfied by most standard queueing formula, e.g., the average delay under M/GI/1 Processor Sharing (PS) queue and the 95th percentile of delay under the M/M/1. Further, the convexity of f_i in m_i models the law of diminishing returns for parallelism.

Combining the above gives the following model for the total delay cost $\mathcal{D}_i(t)$ at data center i during timeslot t:

$$\mathcal{D}_i(t) = \sum\nolimits_{j \in J} \lambda_{ij}(t) r\Big(f_i(m_i(t), \lambda_i(t)) + d_{ij}(t)\Big). \quad (2)$$

C. The geographical load balancing problem

Given the cost models above, the goal of geographical load balancing is to choose the routing policy $\lambda_{ij}(t)$ and the number of active servers in each data center $m_i(t)$ at each time t in order minimize the total cost during $[1, T]$. Because this model neglects the cost of turning servers on and off, the optimization decouples into independent sub-problems for each timeslot t. For the analysis we consider only a single interval.[3] Thus the minimization of the aggregate of $\mathcal{E}_i(t) + \mathcal{D}_i(i)$ is achieved by solving, at each timeslot,

$$\min_{\mathbf{m}, \boldsymbol{\lambda}} \quad \sum_{i \in N} g_i(m_i, \lambda_i) + \sum_{i \in N} \sum_{j \in J} \lambda_{ij} r(d_{ij} + f_i(m_i, \lambda_i)) \quad (3a)$$

$$\text{s.t.} \quad \sum\nolimits_{i \in N} \lambda_{ij} = L_j, \qquad \forall j \in J \quad (3b)$$

$$\lambda_{ij} \geq 0, \qquad \forall i \in N, \forall j \in J \quad (3c)$$

$$0 \leq m_i \leq M_i, \qquad \forall i \in N. \quad (3d)$$

where $\mathbf{m} = (m_i)_{i \in N}$ and $\boldsymbol{\lambda} = (\lambda_{ij})_{i \in N, j \in J}$. We refer to this formulation as GLB. Note that GLB is jointly convex in λ_{ij} and m_i and can be efficiently solved centrally [23]. However, a distributed solution algorithm is usually required by large-scale systems, such as those derived in Section IV.

In contrast to prior work on geographical load balancing, this paper jointly optimizes total energy cost and end-to-end

[1] Minimizing server energy consumption also reduces cooling and power distribution costs.

[2] We focus on the issue of peak pricing in our recent work [22]. It requires slightly different approaches, but they can be merged.

[3] Time-dependence of L_j and prices is re-introduced for, and central to, the numeric results in Sections V and VI.

user delay, with consideration of both price and network delay diversity. To our knowledge, this is the first work to do so.

GLB provides a general framework for studying geographical load balancing. However, the model still ignores many aspects of data center design, e.g., reliability and availability, which are central to data center service level agreements. Such issues are beyond the scope of this paper; however our designs merge nicely with proposals such as [24] for these goals.

The GLB model is too broad for some of our analytic results and thus we often use two restricted versions.

Linear lost revenue. There is evidence that lost revenue is linear within the range of interest for sites such as Google, Bing, and Shopzilla [25], [26]. To model this, we can let $r(d) = \beta d$, for constant β. GLB then simplifies to

$$\min_{\mathbf{m},\boldsymbol{\lambda}} \sum_{i\in N} g_i(m_i,\lambda_i) + \beta \left(\sum_{i\in N} \lambda_i f_i(m_i,\lambda_i) + \sum_{i\in N}\sum_{j\in J} d_{ij}\lambda_{ij} \right) \tag{4}$$

subject to (3b)–(3d). We call this optimization GLB-LIN.

Queueing-based delay. We occasionally specify the form of f and g using queueing models. This provides increased intuitions about the distributed algorithms presented.

If the workload is perfectly parallelizable, and arrivals are Poisson, then $f_i(m_i,\lambda_i)$ is the average delay of m_i parallel queues, with arrival rate λ_i/m_i. Moreover, if each queue is an M/GI/1 PS queue, $f_i(m_i,\lambda_i) = 1/(\mu_i - \lambda_i/m_i)$. We also assume $g_i(m_i,\lambda_i) = p_i m_i$, which implies that the increase in energy cost per timeslot for being in an active state, rather than a low-power state, is m_i regardless of λ_i. Note that cooling efficiency of data center i can be integrated in p_i, which allows incorporation of cooling power consumption.

Under these restrictions, the GLB formulation becomes:

$$\min_{\mathbf{m},\boldsymbol{\lambda}} \sum_{i\in N} p_i m_i + \beta \sum_{j\in J}\sum_{i\in N} \lambda_{ij}\left(\frac{1}{\mu_i - \lambda_i/m_i} + d_{ij}\right) \tag{5a}$$

subject to (3b)–(3d) and the additional constraint

$$\lambda_i \le m_i \mu_i \quad \forall i \in N. \tag{5b}$$

We refer to this optimization as GLB-Q.

Additional Notation. Throughout the paper we use $|S|$ to denote the cardinality of a set S and bold symbols to denote vectors or tuples. In particular, $\boldsymbol{\lambda}_j = (\lambda_{ij})_{i\in N}$ denotes the tuple of λ_{ij} from source j, and $\boldsymbol{\lambda}_{-j} = (\lambda_{ik})_{i\in N, k\in J\setminus\{j\}}$ denotes the tuples of the remaining λ_{ik}, which forms a matrix.

We also need the following in discussing the algorithms. Define $F_i(m_i,\lambda_i) = g_i(m_i,\lambda_i) + \beta\lambda_i f_i(m_i,\lambda_i)$, and define $F(\mathbf{m},\boldsymbol{\lambda}) = \sum_{i\in N} F_i(m_i,\lambda_i) + \Sigma_{ij}\lambda_{ij}d_{ij}$. Further, let $\hat{m}_i(\lambda_i)$ be the unconstrained optimal m_i at data center i given fixed λ_i, i.e., the unique solution to $\partial F_i(m_i,\lambda_i)/\partial m_i = 0$.

D. Practical considerations

Our model assumes there exist mechanisms for dynamically (i) provisioning capacity of data centers, and (ii) adapting the routing of requests from sources to data centers. With respect to (i), many dynamic server provisioning techniques are being explored by both academics and industry, e.g., [27]–[30]. With respect to (ii), there are also a variety of protocol-level mechanisms employed for data center selection today. They include, (a) dynamically generated DNS responses, (b) HTTP redirection, and (c) using persistent HTTP proxies to tunnel requests. Each of these has been evaluated thoroughly, e.g., [10], [31]–[33], and though DNS has drawbacks it remains the preferred mechanism for many industry leaders such as Akamai, possibly due to the added latency due to HTTP redirection and tunneling [34]. Within the GLB model, we have implicitly assumed that there exists a proxy/DNS server co-located with each source. The practicality is also shown by [35]. Our model also assumes that the network delays, d_{ij} can be estimated, which has been studied extensively, including work on reducing the overhead of such measurements, e.g., [36], and mapping and synthetic coordinate approaches, e.g., [37], [38]. We discuss the sensitivity of our algorithms to error in these estimates in Section V.

III. Characterizing the optima

We now provide characterizations of the optimal solutions to GLB, which are important for proving convergence of the distributed algorithms in Section IV. They are also necessary because, a priori, one might worry that the optimal solution requires a very complex routing structure, which would be impractical; or that the set of optimal solutions is very fragmented, which would slow convergence in practice. The results here show that such worries are unwarranted.

Uniqueness of optimal solution

To begin, note that GLB has at least one optimal solution. This can be seen by applying Weierstrass' theorem [39], since the objective function is continuous and the feasible set is compact subset of $\mathbb{R}^n$. Although the optimal solution is generally not unique, there are natural aggregate quantities unique over the set of optimal solutions, which is a convex set. These are the focus of this section.

A first result is that for the GLB-LIN formulation, under weak conditions on f_i and g_i, we have that λ_i is common across all optimal solutions. Thus, the input to the data center provisioning optimization is unique.

Theorem 1. *Consider the GLB-LIN formulation. Suppose that for all i, $F_i(m_i,\lambda_i)$ is jointly convex in λ_i and m_i, and continuously differentiable in λ_i. Further, suppose that $\hat{m}_i(\lambda_i)$ is strictly convex. Then, for each i, λ_i is common for all optimal solutions.*

The proof is in the Appendix. Note that theorem 1 implies that the server arrival rates at each data center, i.e., λ_i/m_i, are common among all optimal solutions.

Though the conditions on F_i and $\hat{m}_i$ are weak, they do not hold for GLB-Q. In that case, $\hat{m}_i(\lambda_i)$ is linear, and thus not strictly convex. Although the λ_i are not common across all optimal solutions in this setting, the server arrival rates remain common across all optimal solutions.

Theorem 2. *For each data center i, the server arrival rates, λ_i/m_i, are common across all optimal solutions to GLB-Q.*

Sparsity of routing

It would be impractical if the optimal solutions to GLB required that requests from each source were divided up among (nearly) all of the data centers. In general, each λ_{ij} could be non-zero, yielding $|N| \times |J|$ flows of requests from sources to data centers, which would lead to significant scaling issues. Luckily, there is guaranteed to exist an optimal solution with extremely sparse routing. Specifically, we have

Theorem 3. *There exists an optimal solution to GLB with at most $(|N| + |J| - 1)$ of the λ_{ij} strictly positive.*

Though Theorem 3 does not guarantee that every optimal solution is sparse, the proof is constructive. Thus, it provides an approach which allows one to transform any optimal solution into a sparse optimal one.

The following result further highlights the sparsity of the routing: any source will route to at most one data center that is not fully active, i.e., where there exists at least one server in power-saving mode.

Theorem 4. *Consider GLB-Q where power costs p_i are drawn from an arbitrary continuous distribution. If any source $j \in J$ has its requests split between multiple data centers $N' \subseteq N$ in an optimal solution, then, with probability 1, at most one data center $i \in N'$ has $m_i < M_i$.*

IV. Algorithms

We now present three distributed algorithms and prove their convergence. For simplicity we focus on GLB-Q; the approaches are applicable more generally, but become much more complex for richer models.

Since GLB-Q is convex, it can be efficiently solved centrally if all necessary information can be collected at a single point, as may be possible if all the proxies and data centers were owned by the same system with real-time synchronization. However there is a strong case for Internet-scale systems to outsource route selection [10]. To meet this need, the algorithms presented below are decentralized and allow each data center and proxy to optimize based on partial information.

These algorithms seek to fill a notable gap in the growing literature on algorithms for geographical load balancing. Specifically, they have provable optimality guarantees for a performance objective that includes both energy and delay, where route decisions are made using energy price and network propagation delay information. The most closely related work [8] investigates the total electricity cost for data centers in a multi-electricity-market environment. It contains the queueing delay inside the data center (assumed to be a centralized $M/M/1$ queue) but neglects the end-to-end user delay. Conversely, [10] uses a simple, efficient algorithm to coordinate the "replica-selection" decisions for load balancing. Other related works, e.g., [1], [7], [8], either do not provide provable guarantees or ignore diverse network delays and/or prices.

Algorithm 1: Gauss-Seidel iteration

Algorithm 1 is motivated by the observation that GLB-Q is separable in m_i, and, less obviously, also separable in $\boldsymbol{\lambda}_j := (\lambda_{ij}, i \in N)$. This allows all data centers as a group and each proxy j to iteratively solve for optimal $\mathbf{m}$ and $\boldsymbol{\lambda}_j$ in a distributed manner, and communicate their intermediate results. Though distributed, Algorithm 1 requires each proxy to solve an optimization problem.

To highlight the separation between data centers and proxies, we reformulate GLB-Q as:

$$\min_{\boldsymbol{\lambda}_j \in \Lambda_j} \min_{m_i \in \mathcal{M}_i} \sum_{i \in N} \left(p_i m_i + \frac{\beta \lambda_i}{\mu_i - \lambda_i / m_i} \right) + \beta \sum_{i \in N} \sum_{j \in J} \lambda_{ij} d_{ij} \tag{6}$$

$$\mathcal{M}_i := [0, M_i], \Lambda_j := \{\boldsymbol{\lambda}_j | \boldsymbol{\lambda}_j \geq 0, \sum_{i \in N} \lambda_{ij} = L_j, \lambda_i \leq m_i \mu_i\} \tag{7}$$

Since the objective and constraints $\mathcal{M}_i$ and Λ_j are separable, this can be solved separately by data centers i and proxies j.

The iterations of the algorithm are indexed by τ, and are assumed to be fast relative to the timeslots t. Each iteration τ is divided into $|J| + 1$ phases. In phase 0, all data centers i concurrently calculate $m_i(\tau + 1)$ based on their own arrival rates $\lambda_i(\tau)$, by minimizing (6) over their own variables m_i:

$$\min_{m_i \in \mathcal{M}_i} \left(p_i m_i + \frac{\beta \lambda_i(\tau)}{\mu_i - \lambda_i(\tau)/m_i} \right), \quad \forall i \in N. \tag{8}$$

In phase j of iteration τ, proxy j minimizes (6) over its own variable by setting $\boldsymbol{\lambda}_j(\tau+1)$ as the best response to $\mathbf{m}(\tau+1)$ and the most recent values of $\boldsymbol{\lambda}_{-j} := (\boldsymbol{\lambda}_k, k \neq j)$. This works because proxy j depends on $\boldsymbol{\lambda}_{-j}$ only through their aggregate arrival rates at the data centers:

$$\lambda_i(\tau, j) := \sum_{l<j} \lambda_{il}(\tau + 1) + \sum_{l>j} \lambda_{il}(\tau), \quad \forall j \in J. \tag{9}$$

To compute $\lambda_i(\tau, j)$, proxy j need not obtain individual $\lambda_{il}(\tau)$ or $\lambda_{il}(\tau+1)$ from other proxies l. Instead, every data center i measures its local arrival rate $\lambda_i(\tau, j) + \lambda_{ij}(\tau)$ in every phase j of the iteration τ and sends this to proxy j at the beginning of phase j. Then proxy j obtains $\lambda_i(\tau, j)$ by subtracting its own $\lambda_{ij}(\tau)$ from the value received from data center i. This has less overhead than direct messaging.

In summary, Algorithm 1 works as follows (note that the minimization (8) has a closed form). Here, $[x]^a := \min\{x, a\}$.

Algorithm 1. *Starting from a feasible initial allocation $\boldsymbol{\lambda}(0)$ and the associated $\mathbf{m}(\boldsymbol{\lambda}(0))$, let*

$$m_i(\tau + 1) := \left[\left(1 + \frac{1}{\sqrt{p_i/\beta}} \right) \cdot \frac{\lambda_i(\tau)}{\mu_i} \right]^{M_i}, \quad \forall i \in N, \tag{10}$$

$$\boldsymbol{\lambda}_j(\tau + 1) := \arg \min_{\boldsymbol{\lambda}_j \in \Lambda_j} \sum_{i \in N} \frac{\lambda_i(\tau, j) + \lambda_{ij}}{\mu_i - (\lambda_i(\tau, j) + \lambda_{ij})/m_i(\tau + 1)} + \sum\nolimits_{i \in N} \lambda_{ij} d_{ij}. \tag{11}$$

Since GLB-Q generally has multiple optimal $\boldsymbol{\lambda}_j^*$, Algorithm 1 is not guaranteed to converge to one particular optimal solution, i.e., for each proxy j, the allocation $\lambda_{ij}(\tau)$ of job j to data centers i may oscillate among multiple optimal allocations. However, both the optimal cost and the optimal per-server arrival rates to data centers will converge.

Theorem 5. *Let $(\mathbf{m}(\tau), \boldsymbol{\lambda}(\tau))$ be a sequence generated by Algorithm 1 when applied to GLB-Q. Then*

(i) Every limit point of $(\mathbf{m}(\tau), \boldsymbol{\lambda}(\tau))$ is optimal.
(ii) $F(\mathbf{m}(\tau), \boldsymbol{\lambda}(\tau))$ converges to the optimal value.
(iii) The per-server arrival rates $(\lambda_i(\tau)/m_i(\tau), i \in N)$ to data centers converge to their unique optimal values.

The proof of Theorem 5 follows from the fact that Algorithm 1 is a modified Gauss-Seidel iteration. This is also the reason for the requirement that the proxies update sequentially. The details of the proof are in Appendix C.

Algorithm 1 assumes that there is a common clock to synchronize all actions. In practice, updates will likely be asynchronous, with data centers and proxies updating with different frequencies using possibly outdated information. The algorithm generalizes easily to this setting, though the convergence proof is more difficult. The convergence rate of Algorithm 1 in a realistic scenario is illustrated numerically in Section V.

Algorithm 2: Distributed gradient projection

Algorithm 2 reduces the computational load on the proxies. In each iteration, instead of each proxy solving a constrained

minimization (11) as in Algorithm 1, Algorithm 2 takes a single step in a descent direction. Also, while the proxies compute their $\boldsymbol{\lambda}_j(\tau+1)$ sequentially in $|J|$ phases in Algorithm 1, they perform their updates all at once in Algorithm 2.

To achieve this, rewrite GLB-Q as

$$\min_{\boldsymbol{\lambda}_j \in \Lambda_j} \sum_{j \in J} F_j(\boldsymbol{\lambda}) \tag{12}$$

where $F(\boldsymbol{\lambda})$ is the result of minimization of (6) over $m_i \in \mathcal{M}_i$ given λ_i. As explained in the definition of Algorithm 1, this minimization is easy: if we denote the solution to (10) by

$$m_i(\lambda_i) := \left[\left(1 + \frac{1}{\sqrt{p_i/\beta}} \right) \cdot \frac{\lambda_i}{\mu_i} \right]^{M_i} \tag{13}$$

then

$$F(\boldsymbol{\lambda}) := \sum_{i \in N} \left(p_i m_i(\lambda_i) + \frac{\beta \lambda_i}{\mu_i - \lambda_i / m_i(\lambda_i)} \right) + \beta \sum_{i,j} \lambda_{ij} d_{ij}.$$

We now sketch the two key ideas behind Algorithm 2. The first is the standard gradient projection idea: move in the steepest descent direction

$$-\nabla F_j(\boldsymbol{\lambda}) := -\left(\frac{\partial F(\boldsymbol{\lambda})}{\partial \lambda_{1j}}, \cdots, \frac{\partial F(\boldsymbol{\lambda})}{\partial \lambda_{|N|j}} \right)$$

and then project the new point into the feasible set $\prod_j \Lambda_j$ with Λ_j given by (7). The standard gradient projection algorithm will converge if $\nabla F(\boldsymbol{\lambda})$ is Lipschitz over $\prod_j \Lambda_j$. This condition, however, does not hold for our F because of the term $\beta\lambda_i/(\mu_i - \lambda_i/m_i)$. The second idea is to construct a compact and convex subset Λ of the feasible set $\prod_j \Lambda_j$ with the following properties: (i) if the algorithm starts in Λ, it stays in Λ; (ii) Λ contains all optimal allocations; (iii) $\nabla F(\boldsymbol{\lambda})$ is Lipschitz over Λ. The algorithm then projects into Λ in each iteration instead of $\prod_j \Lambda_j$. This guarantees convergence.

Specifically, fix a feasible initial allocation $\boldsymbol{\lambda}(0) \in \prod_j \Lambda_j$ and let $\phi := F(\boldsymbol{\lambda}(0))$ be the initial objective value. Define

$$\Lambda := \Lambda(\phi) := \prod_j \Lambda_j \ \cap \ \left\{ \boldsymbol{\lambda} \,\middle|\, \lambda_i \le \frac{\phi M_i \mu_i}{\phi + \beta M_i}, \ \forall i \right\}. \tag{14}$$

Even though the Λ defined in (14) indeed has the desired properties (see Appendix D), the projection into Λ requires coordination of all proxies and is thus impractical. In order for each proxy j to perform its update in a decentralized manner, we define proxy j's own constraint subset:

$$\hat{\Lambda}_j(\tau) := \Lambda_j \cap \left\{ \boldsymbol{\lambda}_j \,\middle|\, \lambda_i(\tau, -j) + \lambda_{ij} \le \frac{\phi M_i \mu_i}{\phi + \beta M_i}, \forall i \ \right\}$$

where $\lambda_i(\tau, -j) := \sum_{l \neq j} \lambda_{il}(\tau)$ is the arrival rate to data center i, excluding arrivals from proxy j. Even though $\hat{\Lambda}_j(\tau)$ involves $\lambda_i(\tau, -j)$ for all i, proxy j can easily calculate these quantities from data center i's measured arrival rates $\lambda_i(\tau)$, as done in Algorithm 1 in (9) and the discussion thereafter, and does not need to communicate with other proxies. Hence, given $\lambda_i(\tau, -j)$ from data centers i, each proxy can project into $\hat{\Lambda}_j(\tau)$ to compute the next iterate $\boldsymbol{\lambda}_j(\tau+1)$ without the need to coordinate with other proxies.[4] Moreover, if $\boldsymbol{\lambda}(0) \in \Lambda$ then $\boldsymbol{\lambda}(\tau) \in \Lambda$ for all iterations τ.

[4]The projection to the nearest point in $\hat{\Lambda}_j(\tau)$ is defined by $[\boldsymbol{\lambda}]_{\hat{\Lambda}_j(\tau)} := \arg\min_{y \in \hat{\Lambda}_j(\tau)} \|y - \boldsymbol{\lambda}\|_2$.

Algorithm 2. *Starting from a feasible initial allocation $\boldsymbol{\lambda}(0)$ and the associated $\mathbf{m}(\boldsymbol{\lambda}(0))$, each proxy j computes, in each iteration τ:*

$$\mathbf{z}_j(\tau+1) := \left[\boldsymbol{\lambda}_j(\tau) - \gamma_j \left(\nabla F_j(\boldsymbol{\lambda}(\tau)) \right) \right]_{\hat{\Lambda}_j(\tau)}, \quad \forall j \in J, \tag{15}$$

$$\boldsymbol{\lambda}_j(\tau+1) := \frac{|J|-1}{|J|} \boldsymbol{\lambda}_j(\tau) + \frac{1}{|J|} \mathbf{z}_j(\tau+1), \quad \forall j \in J. \tag{16}$$

where $\gamma_j > 0$ is a stepsize.

All data centers i must compute $m_i(\lambda_i(\tau))$ according to (13) in each iteration τ. Each data center i measures the local arrival rate $\lambda_i(\tau)$, calculates $m_i(\lambda_i(\tau))$, and broadcasts these values to all proxies at the beginning of iteration $\tau+1$ for the proxies to compute their $\boldsymbol{\lambda}_j(\tau+1)$.

Algorithm 2 has the same convergence property as Algorithm 1, provided the stepsize is small enough.

Theorem 6. *Let $(\mathbf{m}(\tau), \boldsymbol{\lambda}(\tau))$ be a sequence generated by Algorithm 2 when applied to GLB-Q. If, for all j, $0 < \gamma_j < \min_{i \in N} \beta^2 \mu_i^2 M_i^4 / (|J|(\phi + \beta M_i)^3)$, then*

(i) Every limit point of $(\mathbf{m}(\tau), \boldsymbol{\lambda}(\tau))$ is optimal.
(ii) $F(\mathbf{m}(\tau), \boldsymbol{\lambda}(\tau))$ converges to the optimal value.
(iii) The per-server arrival rates $(\lambda_i(\tau)/m_i(\tau), i \in N)$ to data centers converge to their unique optimal values.

Theorem 6 is proved in Appendix D. The key novelty of the proof is (i) handling the fact that the objective is not Lipshitz and (ii) allowing distributed computation of the projection. The bound on γ_j in Theorem 6 is more conservative than necessary for large systems. Hence, a larger stepsize can be choosen to accelerate convergence. The convergence rate is illustrated in a realistic setting in Section V.

Algorithm 3: Distributed Gradient Descent

Like Algorithm 2, Algorithm 3 is a gradient-based algorithm. The key distinction is that Algorithm 3 avoids the need for projection in each iteration, based on two ideas. First, instead of moving in the steepest descent direction, each proxy j re-distributes its jobs among the datacenters so that $\sum_i \lambda_{ij}(\tau)$ always equals to L_j in each iteration τ. Second, instead of a constant stepsize, Algorithm 3 carefully adjusts a time-varying stepsize in each iteration to ensure that the new allocation is feasible without the need for projection. The design of the stepsize must be such that each proxy j can set its own $\gamma_j(\tau)$ in iteration τ using only local information. Moreover, $\gamma_j(\tau)$ must ensure: (i) collectively $\boldsymbol{\lambda}(\tau+1)$ must stay in the set Λ' over which ∇F is Lipschitz; (ii) $\boldsymbol{\lambda}(\tau+1) \ge 0$; and (iii) $F(\boldsymbol{\lambda}(\tau))$ decreases sufficiently in each iteration. Define

$$\Lambda' := \Lambda'(\phi) = \Pi_j \Lambda_j \cap \left\{ \boldsymbol{\lambda} | \lambda_i \le \frac{\phi + \beta M_i/2}{\phi + \beta M_i} M_i \mu_i, \forall i \right\}$$

Specifically, let ∇_{ij} denote $\partial / \partial \lambda_{ij}$, choose a small $\epsilon \in \left(0, \min_i \left(\frac{\sqrt{p_i/\beta}}{\left(1+\sqrt{p_i/\beta}\right)|J|} M_i \mu_i \right)\right)$, and let

$$\Omega_j(\tau, x) := \{ i | \lambda_{ij}(\tau) > \epsilon \text{ or } \nabla_{ij} F(\boldsymbol{\lambda}(\tau)) < x, \ i \in N \}$$

be the set of data centers that *either* are allocated significant amount of data, i.e., larger than ϵ, from j in round τ *or* will receive an increased allocation from j in round $\tau+1$, i.e., those with a gradient less than x. Then let

$$\left\{ x : \sum_{i \in \Omega_j(\tau,x)} \nabla_{ij} F(\boldsymbol{\lambda}(\tau)) = x | \Omega_j(\tau, x) | \right\} \tag{17}$$

and $\Omega_j(\tau) := \Omega_j(\tau, \theta_j(\tau))$. Note that $i \notin \Omega_j(\tau)$ implies $\lambda_{ij}(\tau) = \lambda_{ij}(\tau+1) \leq \epsilon$.

Let $\Gamma_j^{\downarrow}(\tau) := \{i | \lambda_{ij}(\tau) > \epsilon \text{ and } \nabla_{ij} F(\boldsymbol{\lambda}(\tau)) > \theta_j(\tau)\}$ be the set of data centers which will receive reduced load from j, and $\Gamma_j^{\uparrow}(\tau) := \{i | \nabla_{ij} F(\boldsymbol{\lambda}(\tau)) < \theta_j(\tau)\}$ be the set which will receive increased load. Then, let

$$\gamma_j^{\downarrow}(\tau) = \min_{i \in \Gamma_j^{\downarrow}(\tau)} \left\{ \frac{\lambda_{ij}(\tau)}{\nabla_{ij} F(\boldsymbol{\lambda}(\tau)) - \theta_j(\tau)} \right\}$$

be the maximum step size for which no data center will be reduced to an allocation below 0 and

$$\gamma_j^{\uparrow}(\tau) = \frac{1}{|J|} \min_{i \in \Gamma_j^{\uparrow}(\tau)} \left\{ \frac{\frac{\phi + \beta M_i/2}{\phi + \beta M_i} M_i \mu_i - \lambda_i(\tau)}{\theta_j(\tau) - \nabla_{ij} F(\boldsymbol{\lambda}(\tau))} \right\}$$

be a lower bound on the maximum step size for which no data center will have its load increased beyond that permitted by $\Lambda_j'(\tau)$. Algorithm 3 proceeds as follows.

Algorithm 3. *Let* $K' = \max_i \frac{16|J|(\phi+\beta M_i)^3}{\beta^2 M_i^4 \mu_i^2}$. *Select* $\varrho \in (0, 2)$. *Starting from a feasible initial allocation* $\boldsymbol{\lambda}(0)$, *each proxy* j *computes, in each iteration* τ*:*

$$\gamma_j(\tau) := \min \left\{ \gamma_j^{\downarrow}(\tau), \gamma_j^{\uparrow}(\tau), \varrho / K' \right\}, \tag{18}$$

$$\lambda_{ij}(\tau+1) := \begin{cases} \lambda_{ij}(\tau) - \gamma_j(\tau) \left(\nabla_{ij} F(\boldsymbol{\lambda}(\tau)) - \theta_j(\tau) \right) & \text{if } i \in \Omega_j(\tau) \\ \lambda_{ij}(\tau) \leq \epsilon & \text{otherwise} \end{cases} \tag{19}$$

As in the case of Algorithm 2, implicit in the description is the requirement that all data centers i compute $m_i(\lambda_i(\tau))$ according to (13) in each iteration τ. The procedure for this is the same as discussed for Algorithm 2.

Theorem 7. *When using Algorithm 3 in the GLB-Q formulation,* $F(\boldsymbol{\lambda}(\tau))$ *converges to a value no greater than optimal value plus* $B\epsilon$, *where* $B = \beta |J| \sum_i \left(\left(1 + \sqrt{p_i/\beta} \right)^2 / \mu_i + 2 \max_j d_{ij} \right)$.

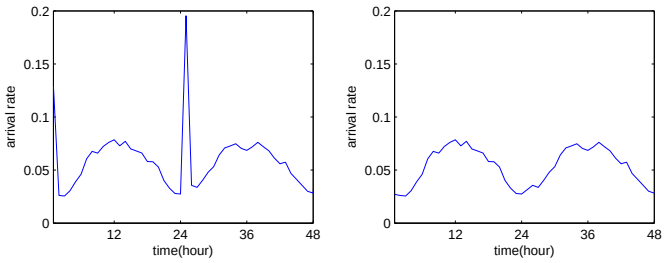
Also, as with Algorithm 2, the key novelty of the proof of Theorem 7 is the fact that we can prove convergence even though the objective function is not Lipschitz. The proof of Theorem 7 is provided in Appendix E. Finally, note that the convergence rate of Algorithm 3 is even faster than that of Algorithm 2 in realistic settings, as we illustrate in Section V. We can consider ϵ as the tolerant of error when the optimal allocation $\lambda_{ij} = 0$. In practice, we can set ϵ to a small value, then Algorithm 3 will *almost* converge to the optimal value.

V. Case Study

The remainder of the paper evaluates the algorithms presented in the previous section under a realistic workload. This section considers the data center perspective (i.e., cost minimization) and Section VI considers the social perspective (i.e., brown energy usage).

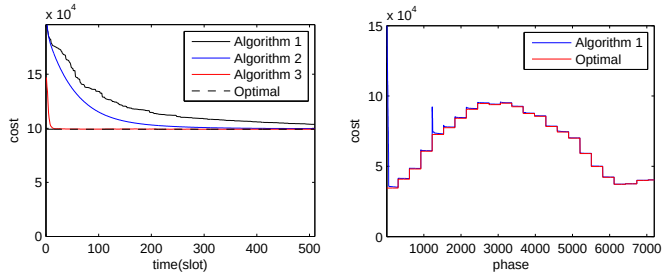
A. Experimental setup

We aim to use realistic parameters in the experimental setup and provide conservative estimates of the cost savings resulting from optimal geographical load balancing. The setup models an Internet-scale system such as Google within the United States.



(a) Original trace (b) Adjusted trace

Fig. 1. Hotmail trace used in numerical results.



(a) Static setting (b) Dynamic setting

Fig. 2. Convergence of all three algorithms.

1) Workload description: To build our workload, we start with a trace of traffic from Hotmail, a large Internet service running on tens of thousands of servers. The trace represents the I/O activity from 8 servers over a 48-hour period, starting at midnight (PDT) on August 4, 2008, averaged over 10 minute intervals. The trace has strong diurnal behavior and has a fairly small peak-to-mean ratio of 1.64. Results for this small peak-to-mean ratio provide a lower bound on the cost savings under workloads with larger peak-to-mean ratios. As illustrated in Figure 1(a), the Hotmail trace contains significant nightly activity due to maintenance processes; however the data center is provisioned for the peak foreground traffic. This creates a dilemma about whether to include the maintenance activity or not. We have performed experiments with both, but report only the results with the spike removed (as illustrated in Figure 1(b)) because this leads to a more conservative estimate of the cost savings. Building on this trace, we construct our workload by placing a source at the geographical center of each US state, co-located with a proxy or DNS server (as described in Section II-D). The trace is shifted according to the time-zone of each state, and scaled by the size of the population in the state that has an Internet connection [40].

2) Data center description: To model an Internet-scale system, we have 14 data centers, one at the geographic center of each state known to have Google data centers [41]: California, Washington, Oregon, Illinois, Georgia, Virginia, Texas, Florida, North Carolina, and South Carolina.

We merge the data centers in each state and set M_i proportional to the number of data centers in that state, while keeping $\Sigma_{i \in N} M_i \mu_i$ twice the total peak workload, $\max_t \Sigma_{j \in J} L_j(t)$. The network delays, d_{ij}, between sources and data centers are taken to be proportional to the geographical distances between them and comparable to the average queueing delays inside the data centers. This lower bound on the network delay ignores delay due to congestion or indirect routes.

3) Cost function parameters: To model the costs of the system, we use the GLB-Q formulation. We set $\mu_i = 1$ for all i, so that the servers at each location are equivalent. We assume

the energy consumption of an active server in one timeslot is normalized to 1. We set constant electricity prices using the industrial electricity price of each state in May 2010 [42]. Specifically, the price (cents per kWh) is 10.41 in California; 3.73 in Washington; 5.87 in Oregon, 7.48 in Illinois; 5.86 in Georgia; 6.67 in Virginia; 6.44 in Texas; 8.60 in Florida; 6.03 in North Carolina; and 5.49 in South Carolina. In this section, we set $\beta = 1$ according to the estimates in [26]; however Figure 3 illustrates the impact of varying β.

4) Algorithm benchmarks: To provide benchmarks for the performance of the algorithms presented here, we consider three baselines, which are approximations of common approaches used in Internet-scale systems. They also allow implicit comparisons with prior work such as [8]. The approaches use different amounts of information to perform the cost minimization. Note that each approach must use queueing delay (or capacity information); otherwise the routing may lead to instability.

Baseline 1 uses network delays but ignores energy price when minimizing its costs. This demonstrates the impact of price-aware routing. It also shows the importance of dynamic capacity provisioning, since without using energy cost in the optimization, every data center will keep every server active.

Baseline 2 uses energy prices but ignores network delay. This illustrates the impact of location-aware routing on the data center costs. Further, it allows us to understand the performance improvement of our algorithms compared to those such as [8], [9] that neglect network delays in their formulations.

Baseline 3 uses neither network delay information nor energy price information when performing its cost minimization. Thus, the traffic is routed so as to balance the delays within the data centers. Though naive, designs such as this are still used by systems today; see [43].

B. Performance evaluation

The evaluation of our algorithms and the cost savings due to optimal geographic load balancing will be organized around the following topics.

1) Convergence: We start by considering the convergence of each of the distributed algorithms. Figure 2(a) illustrates the convergence of each of the algorithms in a static setting for t = 11am, where load and electricity prices are fixed and each phase in Algorithm 1 is considered as an iteration. It validates the convergence analysis for both algorithms. Note here Algorithm 2 and Algorithm 3 use a step size $\gamma = 10$; this is much larger than that used in the convergence analysis, which is quite conservative, and there is no sign of causing lack of convergence.

To demonstrate the convergence in a dynamic setting, Figure 2(b) shows Algorithm 1's response to the first day of the Hotmail trace, with loads averaged over one-hour intervals for brevity. One iteration is performed every 10 minutes. This figure shows that even the slower algorithm, Algorithm 1, converges fast enough to provide near-optimal cost. Hence, the remaining plots show only the optimal solution.

2) Energy versus delay tradeoff: The optimization objective we have chosen to model the data center costs imposes a particular tradeoff between the delay and the energy costs, β. It is important to understand the impact of this factor. Figure 3 illustrates how the delay and energy cost trade off under the optimal solution as β changes. Thus, the plot shows the Pareto frontier for the GLB-Q formulation. The figure highlights that there is a smooth convex frontier with a mild 'knee'.

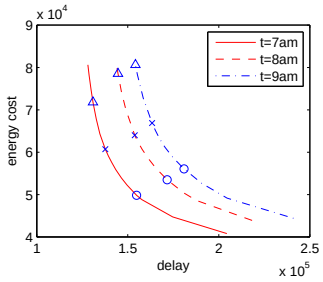


Fig. 3. Pareto frontier of the GLB-Q formulation as a function of β for three different times (and thus arrival rates), PDT. Circles, x-marks, and triangles correspond to $\beta = 0.4$, 1, and 2.5, respectively.

3) Cost savings: To evaluate the cost savings of geographical load balancing, Figure 4 compares the optimal costs to those incurred under the three baseline strategies described in the experimental setup. The overall cost, shown in Figures 4(a) and 4(b), is significantly lower under the optimal solution than all of the baselines (nearly 40% during times of light traffic). Recall that Baseline 2 is the state of the art, studied in recent papers such as [8], [9].

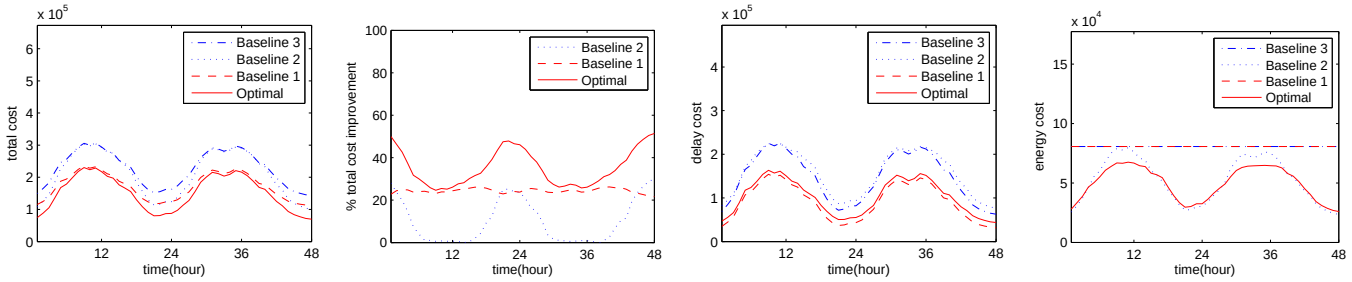
To understand where the benefits are coming from, let us consider separately the two components of cost: delay and energy. Figures 4(c) and 4(d) show that the optimal algorithm performs well with respect to both delay and energy costs individually. In particular, Baseline 1 provides a lower bound on the achievable delay costs, and the optimal algorithm nearly matches this lower bound. Similarly, Baseline 2 provides a natural bar for comparing the achievable energy cost. At periods of light traffic the optimal algorithm provides nearly the same energy cost as this baseline, and (perhaps surprisingly) during periods of heavy-traffic the optimal algorithm provides significantly lower energy costs. The explanation for this is that, when network delay is considered by the optimal algorithm, if all the close data centers have all servers active, a proxy might still route to them; however when network delay is not considered, a proxy is more likely to route to a data center that is not yet running at full capacity, thereby adding to the energy cost.

4) Sensitivity analysis: We have studied the sensitivity of the algorithms to errors in the inputs load L_j and network delay d_{ij}. Estimation errors in L_j only affect the routing. In our model the data centers adapt their number of servers based on the true load, which counteracts suboptimal routing. In our context, network delay was 15% of the cost, and so large relative errors in delay had little impact. Baseline 2 can be thought of as applying the optimal algorithm to extremely poor estimates of d_{ij} (namely $d_{ij} = 0$), and so the Figure 4(a) provides some illustration of the effect of estimation error.

VI. Social Impact

We now shift focus from the cost savings of the data center operator to the social impact of geographical load balancing. We focus on the impact of geographical load balancing on the usage of "brown" non-renewable energy by Internet-scale systems, and how this impact depends on electricity pricing.

Intuitively, geographical load balancing allows the traffic to "follow the renewables"; thus providing increased usage of green energy and decreased brown energy usage. However, such benefits are only possible if data centers forgo static energy contracts for dynamic energy pricing (either through demand response programs or real-time pricing). The experiments in this section show that if dynamic pricing is



(a) Total cost (b) Total cost (% improvement vs. Baseline 3) (c) Delay cost (d) Energy cost

Fig. 4. Impact of ignoring network delay and/or energy price on the cost incurred by geographical load balancing.

done optimally, then geographical load balancing can provide significant social benefits by reducing non-renewable energy consumption.

A. Experimental setup

To explore the social impact of geographical load balancing, we use the setup described in Section V. However, we add models for the availability of renewable energy, the pricing of renewable energy, and the social objective.

1) The availability of renewable energy: To capture the availability of wind and solar energy, we use traces of wind speed and Global Horizontal Irradiance (GHI) obtained from [44], [45] that have measurements every 10 minutes for a year. The normalized generations of four states (CA, TX, IL, NC) and the West/East Coast average are illustrated in Figure 5(a), where 50% of renewable energy comes from solar.

Building on these availability traces, for each location we let $\alpha_i(t)$ denote the fraction of the energy that is from renewable sources at time t, and let $\bar{\alpha} = (|N|\,T)^{-1}\sum_{t=1}^{T}\sum_{i\in N}\alpha_i(t)$ be the "penetration" of renewable energy. We take $\bar{\alpha} = 0.30$, which is on the progressive side of the renewable targets among US states [46].

Finally, when measuring the brown/green energy usage of a data center at time t, we use simply $\sum_{i\in N}\alpha_i(t)m_i(t)$ as the green energy usage and $\sum_{i\in N}(1-\alpha_i(t))m_i(t)$ as the brown energy usage. This models the fact that the grid cannot differentiate the source of the electricity provided.

2) Dynamic pricing and demand response: Internet-scale systems have spatial flexibility in energy usage that is not available to traditional energy consumers; thus they are well positioned to take advantage of demand response and real-time pricing to reduce both their electricity costs and their brown energy consumption.

To provide a simple model of demand response, we use time-varying prices $p_i(t)$ in each time-slot that depend on the availability of renewable resources $\alpha_i(t)$ in each location.

The way $p_i(t)$ is chosen as a function of $\alpha_i(t)$ will be of fundamental importance to the social impact of geographical load balancing. To highlight this, we consider a parameterized "differentiated pricing" model that uses a price p_b for brown energy and a price p_g for green energy. Specifically,

$$p_i(t) = p_b(1-\alpha_i(t)) + p_g\alpha_i(t).$$

Note that $p_g = p_b$ corresponds to static pricing, and we show in the next section that $p_g = 0$ corresponds to socially optimal pricing. Our experiments vary $p_g \in [0, p_b]$. p_b is the same price as used in the previous section.

3) The social objective: To model the social impact of geographical load balancing we need to formulate a social objective. Like the GLB formulation, this must include a tradeoff between the energy usage and the average delay users of the system experience, because purely minimizing brown energy use requires all $m_i = 0$. The key difference between the GLB formulation and the social formulation is that the *cost* of energy is no longer relevant. Instead, the environmental impact is important, and thus the brown energy usage should be minimized. This leads to the following simple model for the social objective:

$$\min_{\mathbf{m}(t),\boldsymbol{\lambda}(t)} \sum_{t=1}^{T}\sum_{i\in N}\left((1-\alpha_i(t))\frac{\mathcal{E}_i(t)}{p_i(t)} + \tilde{\beta}\mathcal{D}_i(t)\right) \qquad (20)$$

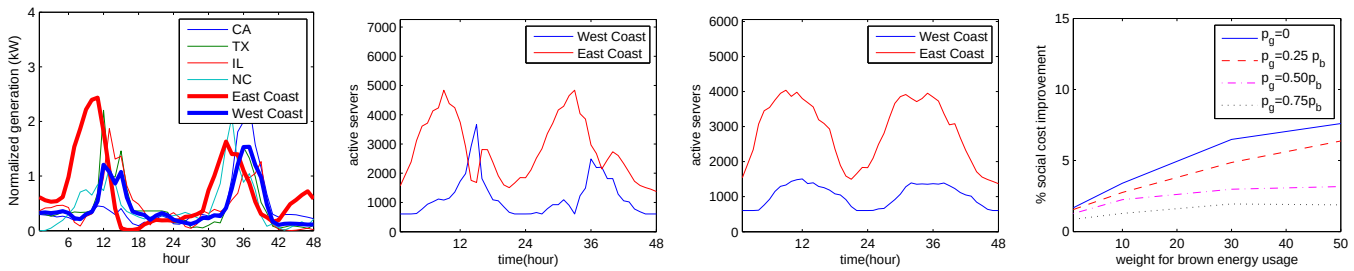
where $\mathcal{D}_i(t)$ is the delay cost defined in (2), $\mathcal{E}_i(t)$ is the energy cost defined in (1), and $\tilde{\beta}$ is the relative valuation of delay versus energy. Further, we have imposed that the energy cost follows from the pricing of $p_i(t)$ cents/kWh in timeslot t. Note that, though simple, our choice of $\mathcal{D}_i(t)$ to model the disutility of delay to users is reasonable because lost revenue captures the lack of use as a function of increased delay.

An immediate observation about the above social objective is that to align the data center and social goals, one needs to set $p_i(t) = (1-\alpha_i(t))/\tilde{\beta}$, which corresponds to choosing $p_b = 1/\tilde{\beta}$ and $p_g = 0$ in the differentiated pricing model above. We refer to this as the "optimal" pricing model.

B. The importance of dynamic pricing

To begin our experiments, we illustrate that optimal pricing can lead geographical load balancing to "follow the renewables." Figure 5 shows this using the renewable traces shown in Figure 5(a). By comparing Figures 5(b) to Figure 5(c), which uses static pricing, the change in capacity provisioning, and thus energy usage, is evident. For example, Figure 5(b) shows a clear shift of service capacity from the west coast to the east coast as solar energy becomes highly available in the east coast and then switch back when solar energy is less available in the east coast but high in the west coast. Though not explicit in the figures, this "follow the renewables" routing has the benefit of significantly reducing the brown energy usage since energy use is more correlated with the availability of renewables. Thus, geographical load balancing provides the opportunity to aid the incorporation of renewables into the grid.

Figure 5 assumed the optimal dynamic pricing, but currently data centers negotiate fixed price contracts. Although there are many reasons why grid operators will encourage data center operators to transfer to dynamic pricing over the coming years,



(a) Renewable availability (b) Optimal dynamic pricing (c) Static pricing (d) Social objective improvement

Fig. 5. Geographical load balancing "following the renewables". (a) Renewable availability. (b) and (c): Capacity provisionings of east coast and west coast data centers when there are renewables, under (b) optimal dynamic pricing and (c) static pricing. (d) Reduction in social cost from dynamic pricing compared to static pricing as a function of the weight for brown energy usage, $1/\tilde{\beta}$, and $\tilde{\beta} = 0.1$.

this is likely to be a slow process [13]. Thus, it is important to consider the impact of partial adoption of dynamic pricing in addition to full, optimal dynamic pricing. Figure 5(d) focuses on this issue. To model the partial adoption of dynamic pricing, we can consider $p_g \in [0, p_b]$. This figure shows that the benefits provided by dynamic pricing are moderate but significant, even at partial adoption (high p_g). Another interesting observation about Figure 5(d) is that the curves increase faster in the range when $1/\tilde{\beta}$ is small, which highlights that the social benefit of geographical load balancing becomes significant even when there is only moderate importance placed on energy. When p_g is higher than p_b, which is common currently, the cost increases and geographical load balancing can no longer help to reduce non-renewable energy consumption. We omit the results due to space constraints. For more recent results about geographical load balancing in Internet-scale systems with local renewable generation and data center demand response to utility coincident peak charging, please refer to [22], [47]

VII. Concluding Remarks

This paper has focused on understanding algorithms for and social impacts of geographical load balancing in Internet-scaled systems. We have provided three distributed algorithms that provably compute the optimal routing and provisioning decisions for Internet-scale systems and we have evaluated these algorithms using trace-based numerical simulations. Further, we have studied the feasibility and benefits of providing demand response for the grid via geographical load balancing. Our experiments highlight that geographical load balancing can provide an effective tool for demand response: when pricing is done carefully, electricity providers can motivate Internet-scale systems to "follow the renewables" and route to areas where green renewable energy is available. This both eases the incorporation of non-dispatchable renewables into the grid and reduces brown energy consumption of Internet-scale systems.

While we have more recent results about online algorithms for geographical load balancing [14] and workload management for local data center [18], there are a number of interesting directions for future work. With respect to the design of distributed algorithms, one aspect that our model has ignored is the switching cost (in terms of delay and wear-and-tear) associated with switching servers into and out of power-saving modes. Our model also ignores issues related to reliability and availability, which are quite important in practice. With respect to the social impact of geographical load balancing, our results highlight the opportunity provided by geographical load balancing for demand response; however there are many issues left to be considered. For example, which demand response market should Internet-scale systems participate in to minimize costs? How can policy decisions such as cap-and-trade be used to provide the proper incentives for Internet-scale systems, such as [6]? Can Internet-scale systems use energy storage at data centers in order to magnify cost reductions when participating in demand response markets? Answering these questions will pave the way for greener geographic load balancing.

Acknowledgments

This work was supported by NSF grants CCF 0830511, CNS 0911041, and CNS 0846025, DoE grant DE-EE0002890, ARO MURI grant W911NF-08-1-0233, Microsoft Research, Bell Labs, the Lee Center for Advanced Networking, and ARC grants FT0991594 and DP130101378.

References

[1] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," in *Proc. ACM Sigcomm*, Aug. 2009.

[2] A. Beloglazov, R. Buyya, Y. C. Lee, A. Zomaya *et al.*, "A taxonomy and survey of energy-efficient data centers and cloud computing systems," *Advances in Computers*, vol. 82, no. 2, pp. 47–111, 2011.

[3] O. S. Unsal and I. Koren, "System-level power-aware deisgn techniques in real-time systems," *Proc. IEEE*, vol. 91, no. 7, pp. 1055–1069, 2003.

[4] S. Irani and K. R. Pruhs, "Algorithmic problems in power management," *SIGACT News*, vol. 36, no. 2, pp. 63–76, 2005.

[5] S. Kaxiras and M. Martonosi, *Computer Architecture Techniques for Power-Efficiency*. Morgan & Claypool, 2008.

[6] K. Le, R. Bianchini, T. D. Nguyen, O. Bilgir, and M. Martonosi, "Capping the brown energy consumption of internet services at low cost," in *Proc. IGCC*, 2010.

[7] E. Pakbaznia and M. Pedram, "Minimizing data center cooling and server power costs," in *Proc. ISLPED*, 2009.

[8] 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 *INFOCOM*, 2010.

[9] R. Stanojevic and R. Shorten, "Distributed dynamic speed scaling," in *Proc. IEEE INFOCOM*, 2010.

[10] P. Wendell, J. W. Jiang, M. J. Freedman, and J. Rexford, "Donar: decentralized server selection for cloud services," in *Proc. ACM Sigcomm*, 2010, pp. 231–242.

[11] "Server and data center energy efficiency," Final Report to Congress, U.S. Environmental Protection Agency, 2007.

[12] Www.enernoc.com.

[13] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's not easy being green," in *Proc. of ACM SIGCOMM 2012*.

[14] M. Lin, Z. Liu, A. Wierman, and L. L. Andrew, "Online algorithms for geographical load balancing," in *Proc. of IGCC*, 2012.

[15] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," in *Proc. IEEE INFOCOM*, 2011.

[16] A. Shah, C. Bash, M. Arlitt, Y. Chen, D. Gmach, R. Sharma, and C. Patel, "Thermal management considerations for geographically distributed computing infrastructures." ASME, 2010.

[17] K. Le, R. Bianchini, J. Zhang, Y. Jaluria, J. Meng, and T. D. Nguyen, "Reducing electricity cost through virtual machine placement in high performance computing clouds," in *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*. ACM, 2011, p. 22.

[18] Z. Liu, Y. Chen, C. Bash, A. Wierman, D. Gmach, Z. Wang, M. Marwah, and C. Hyser, "Renewable and cooling aware workload management for sustainable data centers," in *Proc. ACM SIGMETRICS 2012*, vol. 40, no. 1. ACM, 2012, pp. 175–186.
[19] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *Proc. ISCA*, 2007.
[20] A. Wierman, L. L. H. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in *Proc. IEEE INFOCOM*, 2009.
[21] L. L. H. Andrew, M. Lin, and A. Wierman, "Optimality, fairness and robustness in speed scaling designs," in *Proc. ACM SIGMETRICS*, 2010.
[22] Z. Liu, A. Wierman, Y. Chen, B. Razon, and N. Chen, "Data center demand response: Avoiding the coincident peak via workload shifting and local generation," in *Proceedings of ACM SIGMETRICS*, 2013.
[23] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.
[24] E. Thereska, A. Donnelly, and D. Narayanan, "Sierra: a power-proportional, distributed storage system," Microsoft Research, Tech. Rep. MSR-TR-2009-153, 2009.
[25] A. Croll and S. Power, "How web speed affects online business KPIs," 2009, http://www.watchingwebsites.com.
[26] Http://perspectives.mvdirona.com/2009/10/31/TheCostOfLatency.aspx.
[27] S. Albers, "Energy-efficient algorithms," *Comm. of the ACM*, vol. 53, no. 5, pp. 86–96, 2010.
[28] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, "Managing server energy and operational costs in hosting centers," in *Proc. ACM SIGMETRICS*, 2005.
[29] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy, "Optimal power allocation in server farms," in *Proc. ACM SIGMETRICS*, 2009.
[30] R. Urgaonkar, U. C. Kozat, K. Igarashi, and M. J. Neely, "Dynamic resource allocation and power management in virtualized data centers," in *IEEE NOMS*, Apr. 2010.
[31] M. Conti and C. Nazionale, "Load distribution among replicated web servers: A QoS-based approach," in *Proc. ACM WISP*, 1999.
[32] Z. M. Mao, C. D. Cranor, F. Bouglis, M. Rabinovich, O. Spatscheck, and J. Wang, "A precise and efficient evaluation of the proximity between web clients and their local DNS servers," in *USENIX*, 2002, pp. 229–242.
[33] M. Pathan, C. Vecchiola, and R. Buyya, "Load and proximity aware request-redirection for dynamic load distribution in peering CDNs," in *Proc. OTM*, 2008.
[34] J. Pang, A. Akella, A. Shaikh, B. Krishnamurthy, and S. Seshan, "On the responsiveness of DNS-based network control," in *Proc. IMC*, 2004.
[35] G. Ghatikar, V. Ganti, N. Matson, and M. Piette, "Demand response opportunities and enabling technologies for data centers: Findings from field studies," 2012.
[36] W. Theilmann and K. Rothermel, "Dynamic distance maps of the internet," in *Proc. IEEE INFOCOM*, 2001.
[37] R. Krishnan, H. V. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao, "Moving beyond end-to-end path information to optimize CDN performance," in *Proc. ACM Sigcomm*, 2009.
[38] E. Ng and H. Zhang, "Predicting internet network distance with coordinates-based approaches," in *Proc. IEEE INFOCOM*, 2002.
[39] D. P. Bertsekas, *Nonlinear Programming*. Athena Scientific, 1999.
[40] "US Census Bureau, http://www.census.gov."
[41] http://www.datacenterknowledge.com, 2008.
[42] http://www.eia.doe.gov.
[43] V. K. Adhikari, S. Jain, and Z.-L. Zhang, "YouTube traffic dynamics and its interplay with a tier-1 ISP: An ISP perspective," in *Proc. ACM IMC*, 2010, pp. 431–443.
[44] http://rredc.nrel.gov, 2010.
[45] http://wind.nrel.gov, 2010.
[46] S. Carley, "State renewable energy electricity policies: An empirical evaluation of effectiveness," *Energy Policy*, vol. 37, no. 8, pp. 3071–3081, Aug 2009.
[47] Z. Liu, M. Lin, A. Wierman, S. Low, and L. Andrew, "Geographical load balancing with renewables," *ACM SIGMETRICS Performance Evaluation Review*, vol. 39, no. 3, pp. 62–66, 2011.
[48] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*. Athena Scientific, 1989.

APPENDIX

We now prove the results from Section III, beginning with the illuminating Karush-Kuhn-Tucker (KKT) conditions.

A. Optimality conditions

As GLB-Q is convex and satisfies Slater's condition, the KKT conditions are necessary and sufficient for optimality [23]; for the other models they are merely necessary.

GLB-Q: Let $\underline{\omega}_i \geq 0$ and $\bar{\omega}_i \geq 0$ be Lagrange multipliers corresponding to (3d), and $\delta_{ij} \geq 0$, ν_j and σ_i be those for (3c), (3b) and (5b). The Lagrangian is then

$$\begin{aligned}\mathcal{L} = &\sum_{i\in N} m_i p_i + \beta \sum_{j\in J}\sum_{i\in N}\left(\frac{\lambda_{ij}}{\mu_i - \lambda_i/m_i} + \lambda_{ij} d_{ij}\right) \\ &- \sum_{i\in N}\sum_{j\in J}\delta_{ij}\lambda_{ij} + \sum_{j\in J}\nu_j\left(L_j - \sum_{i\in N}\lambda_{ij}\right) \\ &+ \sum_{i\in N}\left(\bar{\omega}_i(m_i - M_i) - \underline{\omega}_i m_i\right) + \sum_{i\in N}\sigma_i\left(m_i\mu_i - \lambda_i\right)\end{aligned}$$

The KKT conditions of stationarity, primal and dual feasibility and complementary slackness are:

$$\beta\left(\frac{\mu_i}{(\mu_i - \lambda_i/m_i)^2} + d_{ij}\right) - \nu_j - \delta_{ij} - \sigma_i = 0, \quad \forall i, j \tag{21}$$

$$\delta_{ij}\lambda_{ij} = 0; \quad \delta_{ij} \geq 0, \quad \lambda_{ij} \geq 0, \quad \forall i, j \tag{22}$$

$$\sigma_i\left(m_i\mu_i - \lambda_i\right) = 0; \quad \sigma_i \geq 0, \quad m_i\mu_i - \lambda_i \geq 0, \quad \forall i \tag{23}$$

$$\sum_{i\in N}\lambda_{ij} = L_j, \quad \forall j \tag{24}$$

$$p_i - \beta\left(\frac{\lambda_i/m_i}{\mu_i - \lambda_i/m_i}\right)^2 + \bar{\omega}_i - \underline{\omega}_i + \sigma_i\mu_i = 0, \quad \forall i \tag{25}$$

$$\bar{\omega}_i(m_i - M_i) = 0; \quad \bar{\omega}_i \geq 0, \quad m_i \leq M_i, \quad \forall i \tag{26}$$

$$\underline{\omega}_i m_i = 0; \quad \underline{\omega}_i \geq 0, \quad m_i \geq 0, \quad \forall i. \tag{27}$$

The conditions (21)–(24) determine the sources' choice of λ_{ij}, and we claim they imply that source j will only send data to those data centers i which have minimum marginal cost $d_{ij} + (1 + \sqrt{p_i^*/\beta})^2/\mu_i$, where $p_i^* = p_i - \underline{\omega}_i + \bar{\omega}_i$. To see this, let $\bar{\lambda}_i = \lambda_i/m_i$. By (25), the marginal queueing delay of data centre i with respect to load λ_{ij} is $\mu_i/(\mu_i - \bar{\lambda}_i)^2 = (1 + \sqrt{p_i^*/\beta})^2/\mu_i$. Thus, from (21), at the optimal point,

$$d_{ij} + \frac{(1 + \sqrt{p_i^*/\beta})^2}{\mu_i} = d_{ij} + \frac{\mu_i}{(\mu_i - \bar{\lambda}_i)^2} = \frac{\nu_j + \delta_{ij}}{\beta} \geq \frac{\nu_j}{\beta} \tag{28}$$

with equality if $\lambda_{ij} > 0$ by (22), establishing the claim.

Note that the solution to (21)–(24) for source j depends on λ_{ik}, $k \neq j$, only through m_i. Given λ_i, data center i finds m_i as the projection onto $[0, M_i]$ of the solution $\hat{m}_i = \lambda_i(1 + \sqrt{p_i/\beta})/(\mu_i\sqrt{p_i/\beta})$ with $\bar{\omega}_i = \underline{\omega}_i = \sigma_i = 0$.

GLB-LIN again decouples into data centers finding m_i given λ_i, and sources finding λ_{ij} given the m_i. Feasibility and complementary slackness conditions (22), (24), (26) and (27) are as for GLB-Q; the stationarity conditions are:

$$\frac{\partial g_i(m_i, \lambda_i)}{\partial \lambda_i} + \beta\left(\frac{\partial\left(\lambda_i f_i(m_i, \lambda_i)\right)}{\partial \lambda_i} + d_{ij}\right) - \nu_j - \delta_{ij} = 0, \quad \forall i, j \tag{29}$$

$$\frac{\partial g_i(m_i, \lambda_i)}{\partial m_i} + \beta\lambda_i\frac{\partial f_i(m_i, \lambda_i)}{\partial m_i} + \bar{\omega}_i - \underline{\omega}_i = 0, \forall i. \tag{30}$$

Note the feasibility constraint (5b) of GLB-Q is no longer explicitly required to ensure stability. In GLB-LIN, it is instead assumed that f is infinite when the load exceeds capacity.

The objective function is strictly convex in data center i's decision variable m_i, and so there is a unique solution $\hat{m}_i(\lambda_i)$ to (30) for $\bar{\omega}_i = \underline{\omega}_i = 0$, and the optimal m_i given λ_i is the projection of this onto the interval $[0, M_i]$.

GLB in its general form has the same KKT conditions as GLB-LIN, with the stationary conditions replaced by

$$\frac{\partial g_i}{\partial \lambda_i} + r(f_i + d_{ij}) + \sum_{k\in J}\lambda_{ik}r'(f_i + d_{ik})\frac{\partial f_i}{\partial \lambda_i} - \nu_j - \delta_{ij} = 0, \forall i, j$$

$$\frac{\partial g_i}{\partial m_i} + \sum_{j\in J}\lambda_{ij}r'(f_i + d_{ij})\frac{\partial f_i}{\partial m_i} + \bar{\omega}_i - \underline{\omega}_i = 0, \quad \forall i$$

where r' denotes the derivative of $r(\cdot)$.

GLB again decouples, since it is convex because $r(\cdot)$ is convex and increasing. However, now data center i's problem depends on all λ_{ij}, rather than simply λ_i.

B. Characterizing the optima

Lemma 8 will help prove the results of Section III.

Lemma 8. *Consider the GLB-LIN formulation. Suppose that for all i, $F_i(m_i, \lambda_i)$ is jointly convex in λ_i and m_i, and differentiable in λ_i where it is finite. If, for some i, the dual variable $\bar{\omega}_i > 0$ for an optimal solution, then $m_i = M_i$ for all optimal solutions. Conversely, if $m_i < M_i$ for an optimal solution, then $\bar{\omega}_i = 0$ for all optimal solutions.*

Proof: Consider an optimal solution S with $i \in N$ such that $\bar{\omega}_i > 0$ and hence $m_i = M_i$. If there exists another optimal solution S' such that $m_i < M_i$. Since the cost function is jointly convex in λ_{ij} and m_i, any convex combination of S and S' must also be optimal. However, since $F_i(m_i, \lambda_i)$ is strictly convex in m_i, the linear combination of S and S' strictly decreases the cost, which contradicts with the fact that S and S' are optimal solutions. ∎

Proof of Theorem 1: Consider first the case where there exists an optimal solution with $m_i < M_i$. By Lemma 8, $\bar{\omega}_i = 0$ for all optimal solutions. Recall that $\hat{m}_i(\lambda_i)$, which defines the optimal m_i, is strictly convex. Thus, if different optimal solutions have different values of λ_i, then a convex combination of the two yielding (m_i', λ_i') would have $\hat{m}_i(\lambda_i') < m_i'$, which contradicts the optimality of m_i'.

Next consider the case where all optimal solutions have $m_i = M_i$. In this case, consider two solutions S and S' that both have $m_i = M_i$. If λ_i is the same under both S and S', we are done. Otherwise, since $F_i(m_i, \lambda_i)$ is strictly convex in λ_i, the linear combination of S and S' strictly decreases the cost, which contradicts with the fact that S and S' are optimal solutions. ∎

Proof of Theorem 2: The proof when $m_i = M_i$ for all optimal solutions is parallel to that of Theorem 1. Otherwise, when $m_i < M_i$ in an optimal solution, the definition of $\hat{m}$ gives $\frac{\lambda_i}{m_i} = \mu_i/(\sqrt{\beta_i/p_i} + 1)$ for all optimal solutions. ∎

Proof of Theorem 3: For each optimal solution S, consider an undirected bipartite graph G with a vertex representing each source and each data center and with an edge connecting i and j when $\lambda_{ij} > 0$. We will show that at least one of these graphs is acyclic. The theorem then follows since an acyclic graph with X nodes has at most $X - 1$ edges.

To prove that there exists one optimal solution with acyclic graph we will inductively reroute requests in a way that removes cycles while preserving optimality. Suppose G contains a cycle. Let C be a minimal cycle, i.e., no strict subset of C is a cycle, and let C be directed.

Construct a new solution $S(\xi)$ from S by adding ξ to λ_{ij} if $(i,j) \in C$, and subtracting ξ from λ_{ij} if $(j,i) \in C$. Note that this does not change the λ_i. To see that $S(\xi)$ is maintains the optimal cost, first note that the change in the objective function of the GLB between S and $S(\xi)$ is equal to

$$\xi \sum_{(j,i)\in C} \Big(r(d_{ij} + f_i(m_i, \lambda_i)) - r(d_{ji} + f_j(m_j, \lambda_j)) \Big) \quad (31)$$

Next note that the multiplier $\delta_{ij} = 0$ since $\lambda_{ij} > 0$ at S. Further, the condition for stationarity in λ_{ij} can be written as $X_i + r(d_{ij} + f_i(m_i, \lambda_i)) - \nu_j = 0$, where X_i does not depend on the choice of j. Since C is minimal, for each $(i,j) \in C$ where $i \in I$ and $j \in J$ there is exactly one (j', i) with $j' \in J$, and vice versa. Thus,

$$\begin{aligned} 0 &= \sum_{(j,i)\in C} (X_i + r(d_{ij} + f_i(m_i, \lambda_i)) - \nu_j) \\ &\quad - \sum_{(i,j)\in C} (X_i + r(d_{ij} + f_i(m_i, \lambda_i)) - \nu_j) \\ &= \sum_{(j,i)\in C} r(d_{ij} + f_i(m_i, \lambda_i)) - \sum_{(i,j)\in C} r(d_{ij} + f_i(m_i, \lambda_i)). \end{aligned}$$

Hence, by (31) the objective of $S(\xi)$ and S are the same.

To complete the proof, we let $(i^*, j^*) = \arg\min_{(i,j)\in C} \lambda_{ij}$. Then $S(\lambda_{i^*,j^*})$ has $\lambda_{i^*,j^*} = 0$. Thus, $S(\lambda_{i^*,j^*})$ has at least one fewer cycle, since it has broken C. Further, by construction, it is still optimal. ∎

Proof of Theorem 4:

It is sufficient to show that, if $\lambda_{kj}\lambda_{k'j} > 0$ then either $m_k = M_k$ or $m_{k'} = M_{k'}$. Consider a case when $\lambda_{kj}\lambda_{k'j} > 0$.

For a generic i, define $c_i = (1 + \sqrt{p_i/\beta})^2/\mu_i$ as the marginal cost (28) when the Lagrange multipliers $\bar{\omega}_i = \underline{\omega}_i = 0$. Since the p_i are chosen from a continuous distribution, we have that with probability 1

$$c_k - c_{k'} \neq d_{k'j} - d_{kj}. \quad (32)$$

However, (28) holds with equality if $\lambda_{ij} > 0$, and so $d_{kj} + (1 + \sqrt{p_k^*/\beta})^2/\mu_k = d_{k'j} + (1 + \sqrt{p_{k'}^*/\beta})^2/\mu_{k'}$. By the definition of c_i and (32), this implies either $p_k^* \neq p_k$ or $p_{k'}^* \neq p_k$. Hence at least one of the Lagrange multipliers $\underline{\omega}_k$, $\bar{\omega}_k$, $\underline{\omega}_{k'}$ or $\bar{\omega}_{k'}$ must be non-zero. However, $\underline{\omega}_i > 0$ would imply $m_i = 0$ whence $\lambda_{ij} = 0$ by (23), which is false by hypothesis, and so either $\bar{\omega}_k$ or $\bar{\omega}_{k'}$ is non-zero, giving the result by (26). ∎

C. Proofs for Algorithm 1

To prove Theorem 5 we apply a variant of Proposition 3.9 of Ch 3 in [48], which gives that if

(i) $F(\mathbf{m}, \boldsymbol{\lambda})$ is continuously differentiable and convex in the convex feasible region (3b)–(3d);
(ii) Every limit point of the sequence is feasible;
(iii) Given the values of $\boldsymbol{\lambda}_{-j}$ and $\mathbf{m}$, there is a unique minimizer of F with respect to λ_j, and given $\boldsymbol{\lambda}$ there is a unique minimizer of F with respect to $\mathbf{m}$.

Then, every limit point of $(\mathbf{m}(\tau), \boldsymbol{\lambda}(\tau))_{\tau=1,2,\ldots}$ is an optimal solution of GLB-Q.

This differs slightly from [48] in that the requirement that the feasible region be closed is replaced by the feasibility of all limit points, and the requirement of strict convexity with respect to each component is replaced by the existence of a unique minimizer. However, the proof is unchanged.

Proof of Theorem 5: To apply the above to prove Theorem 5, we need to show that $F(\mathbf{m}, \boldsymbol{\lambda})$ satisfies the differentiability and continuity constraints under the GLB-Q model.

GLB-Q is continuously differentiable and, as noted in Appendix A, a convex problem. To see that every limit point is feasible, note that the only infeasible points in the closure of the feasible region are those with $m_i\mu_i = \lambda_i$. Since the objective approaches ∞ approaching that boundary, and Gauss-Seidel iterations always reduce the objective [48], these points cannot be limit points.

It remains to show the uniqueness of the minimum in $\mathbf{m}$ and each $\boldsymbol{\lambda}_j$. Since the cost is separable in the m_i, it is sufficient

to show that this applies with respect to each m_i individually. If $\lambda_i = 0$, then the unique minimizer is $m_i = 0$. Otherwise

$$\frac{\partial^2 F(\mathbf{m},\boldsymbol{\lambda})}{\partial m_i^2} = 2\beta\mu_i \frac{\lambda_i^2}{(m_i\mu_i - \lambda_i)^3}$$

which by (5b) is strictly positive. The Hessian of $F(\mathbf{m},\boldsymbol{\lambda})$ with respect to $\boldsymbol{\lambda}_j$ is diagonal with ith element

$$2\beta\mu_i \frac{m_i^2}{(m_i\mu_i - \lambda_i)^3} > 0$$

which is positive definite except the points where some $m_i = 0$. However, if $m_i = 0$, the unique minimum is $\lambda_{ij} = 0$. Note we cannot have all $m_i = 0$. Except these points, $F(\mathbf{m},\boldsymbol{\lambda})$ is strictly convex in $\boldsymbol{\lambda}_j$ given $\mathbf{m}$ and $\boldsymbol{\lambda}_{-j}$. Therefore $\boldsymbol{\lambda}_j$ is unique given $\mathbf{m}$.

Part (ii) of Theorem 5 follows from part (i) and the continuity of $F(\mathbf{m},\boldsymbol{\lambda})$. Part (iii) follows from part (i) and Theorem 2, which provides the uniqueness of optimal per-server arrival rates $(\lambda_i(\tau)/m_i(\tau), i \in N)$. ■

D. Proofs for Algorithm 2

As discussed in the section on Algorithm 2, we will prove Theorem 6 in three steps. First, we will show that, starting from an initial feasible point $\boldsymbol{\lambda}(0)$, Algorithm 2 generates a sequence $\boldsymbol{\lambda}(\tau)$ that lies in the set $\Lambda := \Lambda(\phi)$ defined in (14), for $\tau = 0, 1, \ldots$. Moreover, $\nabla F(\boldsymbol{\lambda})$ is Lipschitz over Λ. Finally, this implies that $F(\boldsymbol{\lambda}(\tau))$ moves in a descent direction that guarantees convergence.

Lemma 9. *Given an initial point $\boldsymbol{\lambda}(0) \in \prod_j \Lambda_j$, let $\phi := F(\boldsymbol{\lambda}(0))$. Then*

1) $\boldsymbol{\lambda}(0) \in \Lambda := \Lambda(\phi)$;
2) *If $\boldsymbol{\lambda}^*$ is optimal then $\boldsymbol{\lambda}^* \in \Lambda$;*
3) *If $\boldsymbol{\lambda}(\tau) \in \Lambda$, then $\boldsymbol{\lambda}(\tau+1) \in \Lambda$.*

Proof: We claim $F(\boldsymbol{\lambda}) \le \phi$ implies $\boldsymbol{\lambda} \in \Lambda$. This is true because $\phi \ge F(\boldsymbol{\lambda}) \ge \sum_k \frac{\beta\lambda_k}{\mu_k - \lambda_k/m_k(\lambda_k)} \ge \frac{\beta\lambda_i}{\mu_i - \lambda_i/m_i(\lambda_i)} \ge \frac{\beta\lambda_i}{\mu_i - \lambda_i/M_i}, \forall i$. Therefore $\lambda_i \le \frac{\phi}{\phi+\beta M_i} M_i\mu_i, \forall i$. Consequently, the intial point $\boldsymbol{\lambda}(0) \in \Lambda$ and the optimal point $\boldsymbol{\lambda}^* \in \Lambda$ because $F(\boldsymbol{\lambda}^*) \le F(\boldsymbol{\lambda})$.

Next we show that $\boldsymbol{\lambda}(\tau) \in \Lambda$ implies $\mathbf{Z}^j(\tau+1) \in \Lambda$, where $\mathbf{Z}^j(\tau+1)$ is $\boldsymbol{\lambda}(\tau)$ except $\boldsymbol{\lambda}_j(\tau)$ is replaced by $\mathbf{z}_j(\tau)$. This holds because $Z^j_{ik}(\tau+1) = \lambda_{ik}(\tau) \ge 0, \forall k \ne j, \forall i$ and $\sum_i Z^j_{ik}(\tau+1) = \sum_i \lambda_{ik}(\tau) = L_k, \forall k \ne j$. From the definiition of the projection on $\hat{\Lambda}_j(\tau)$, $Z^j_{ij}(\tau+1) \ge 0, \forall i$, $\sum_i Z^j_{ij}(\tau+1) = L_j$, and $\sum_k Z^j_{ik}(\tau+1) \le \frac{\phi}{\phi+\beta M_i} M_i\mu_i, \forall i$. These together ensure $\mathbf{Z}^j(\tau+1) \in \Lambda$.

The update $\boldsymbol{\lambda}_j(\tau+1) = \frac{|J|-1}{|J|}\boldsymbol{\lambda}_j(\tau) + \frac{1}{|J|}\mathbf{z}_j(\tau), \forall j$ is equivalent to $\boldsymbol{\lambda}(\tau+1) = \frac{\sum_j \mathbf{Z}^j(\tau+1)}{|J|}$. Then from the convexity of Λ, we have $\boldsymbol{\lambda}(\tau+1) \in \Lambda$. ■

Let $F(\mathbf{M},\boldsymbol{\lambda})$ be the total cost when all data centers use all servers, and $\nabla F(\mathbf{M},\boldsymbol{\lambda})$ be the derivatives with respect to $\boldsymbol{\lambda}$. To prove that $\nabla F(\boldsymbol{\lambda})$ is Lipschitz over Λ, we need the following intermediate result. We omit the proof due to space constraint.

Lemma 10. *For all $\boldsymbol{\lambda}^a, \boldsymbol{\lambda}^b \in \Lambda$, we have*

$$\left\|\nabla F(\boldsymbol{\lambda}^b) - \nabla F(\boldsymbol{\lambda}^a)\right\|_2 \le \left\|\nabla F(\mathbf{M},\boldsymbol{\lambda}^b) - \nabla F(\mathbf{M},\boldsymbol{\lambda}^a)\right\|_2.$$

Lemma 11. $\left\|\nabla F(\boldsymbol{\lambda}^b) - \nabla F(\boldsymbol{\lambda}^a)\right\|_2 \le K\left\|\boldsymbol{\lambda}^b - \boldsymbol{\lambda}^a\right\|_2$, $\forall \boldsymbol{\lambda}^a, \boldsymbol{\lambda}^b \in \Lambda$, *where* $K = |J| \max_i 2(\phi+\beta M_i)^3/(\beta^2 M_i^4 \mu_i^2)$.

Proof: Following Lemma 10, here we continue to show $\left\|\nabla F(\mathbf{M},\boldsymbol{\lambda}^b) - \nabla F(\mathbf{M},\boldsymbol{\lambda}^a)\right\|_2 \le K\left\|\boldsymbol{\lambda}^b - \boldsymbol{\lambda}^a\right\|_2$.

The Hessian $\nabla^2 F(\mathbf{M},\boldsymbol{\lambda})$ of $F(\mathbf{M},\boldsymbol{\lambda})$ is given by

$$\nabla^2 F_{ij,kl}(\mathbf{M},\boldsymbol{\lambda}) = \begin{cases} \frac{2\beta\mu_i/M_i}{(\mu_i - \lambda_i/M_i)^3} & \text{if } i = k \\ 0 & \text{otherwise.} \end{cases}$$

Then, by the matrix form of Hölder's inequality and the symmetry of $\nabla^2 F(\mathbf{M},\boldsymbol{\lambda})$, we have $\left\|\nabla^2 F\right\|_2^2 \le \left\|\nabla^2 F\right\|_1 \left\|\nabla^2 F\right\|_\infty = \left\|\nabla^2 F\right\|_\infty^2$. Finally, we have

$$\begin{aligned}\left\|\nabla^2 F(\mathbf{M},\boldsymbol{\lambda})\right\|_\infty &= \max_{ij}\left\{\sum_{kl} \nabla^2 F_{ij,kl}(\mathbf{M},\boldsymbol{\lambda})\right\} \\ &= \max_i\left\{|J|\frac{2\beta\mu_i/M_i}{(\mu_i - \lambda_i/M_i)^3}\right\} \le |J| \max_i \frac{2(\phi+\beta M_i)^3}{\beta^2 M_i^4 \mu_i^2}.\end{aligned}$$

In the last step we substitute λ_i by $\frac{\phi M_i\mu_i}{\phi+\beta M_i}$ because $\lambda_i \le \frac{\phi}{\phi+\beta M_i} M_i\mu_i, \forall i$ and $\frac{2\mu_i/M_i}{(\mu_i - \lambda_i/M_i)^3}$ is increasing in λ_i. ■

Lemma 12. *When applying Algorithm 2 to GLB-Q,*

(a) $F(\boldsymbol{\lambda}(\tau+1)) \le F(\boldsymbol{\lambda}(\tau)) - (\frac{1}{\bar{\gamma}_m} - \frac{K}{2})\left\|\boldsymbol{\lambda}(\tau+1) - \boldsymbol{\lambda}(\tau)\right\|_2^2$, *where* $K = |J| \max_i 2(\phi+\beta M_i)^3/(\beta^2 M_i^4\mu_i^2)$, $\bar{\gamma}_m = \max_j \gamma_j$. *Therefore* $F(\boldsymbol{\lambda}(\tau+1)) < F(\boldsymbol{\lambda}(\tau))$ *if* $0 < \bar{\gamma}_m < 2/K$.

(b) $\boldsymbol{\lambda}(\tau+1) = \boldsymbol{\lambda}(\tau)$ *if and only if* $\boldsymbol{\lambda}(\tau)$ *minimizes* $F(\boldsymbol{\lambda})$ *over the set* Λ.

(c) The mapping $T(\boldsymbol{\lambda}(\tau)) = \boldsymbol{\lambda}(\tau+1)$ *is continuous.*

Proof: From the Lemma 11, we know

$$\left\|\nabla F(\boldsymbol{\lambda}^b) - \nabla F(\boldsymbol{\lambda}^a)\right\|_2 \le K\left\|\boldsymbol{\lambda}^b - \boldsymbol{\lambda}^a\right\|_2, \forall \boldsymbol{\lambda}^a \in \Lambda, \forall \boldsymbol{\lambda}^b \in \Lambda$$

where $K = |J| \max_i 2(\phi+\beta M_i)^3/(\beta^2 M_i^4\mu_i^2)$.

Here $\mathbf{Z}^j(\tau+1) \in \Lambda, \boldsymbol{\lambda}(\tau) \in \Lambda$, therefore we have

$$\left\|\nabla F(\mathbf{Z}^j(\tau+1)) - \nabla F(\boldsymbol{\lambda}(\tau))\right\|_2 \le K\left\|\mathbf{Z}^j(\tau+1) - \boldsymbol{\lambda}(\tau)\right\|_2.$$

From the convexity of $F(\boldsymbol{\lambda})$, we have

$$\begin{aligned} F(\boldsymbol{\lambda}(\tau+1)) &= F\left(\frac{\sum_j \mathbf{Z}^j(\tau+1)}{|J|}\right) \\ &\le \frac{1}{|J|}\sum_j F(\mathbf{Z}^j(\tau+1)) \\ &\le \frac{1}{|J|}\sum_j \left(F(\boldsymbol{\lambda}(\tau)) - \left(\frac{1}{\gamma_j} - \frac{K}{2}\right)\left\|\mathbf{Z}^j(\tau+1) - \boldsymbol{\lambda}(\tau)\right\|_2^2\right) \\ &= F(\boldsymbol{\lambda}(\tau)) - \sum_j \left(\frac{1}{\gamma_j} - \frac{K}{2}\right)\frac{\left\|\mathbf{Z}^j(\tau+1) - \boldsymbol{\lambda}(\tau)\right\|_2^2}{|J|} \\ &\le F(\boldsymbol{\lambda}(\tau)) - \left(\frac{1}{\gamma_m} - \frac{K}{2}\right)\frac{\sum_j \left\|\mathbf{Z}^j(\tau+1) - \boldsymbol{\lambda}(\tau)\right\|_2^2}{|J|} \end{aligned}$$

where $K = |J| \max_i 2(\phi+\beta M_i)^3/(\beta^2 M_i^4\mu_i^2)$.

The first line is from the update rule of $\boldsymbol{\lambda}(\tau)$. The second line is from the convexity of $F(\boldsymbol{\lambda})$. The third line is from the property of gradient projection. The last line is from the definition of γ_m.

Then from the convexity of $\|\cdot\|_2^2$, we have

$$\frac{\sum_j \left\|\mathbf{Z}^j(\tau+1)-\boldsymbol{\lambda}(\tau)\right\|_2^2}{|J|} \geq \left\|\frac{\sum_j\left(\mathbf{Z}^j(\tau+1)-\boldsymbol{\lambda}(\tau)\right)}{|J|}\right\|_2^2$$
$$= \left\|\frac{\sum_j \mathbf{Z}^j(\tau+1)}{|J|}-\boldsymbol{\lambda}(\tau)\right\|_2^2 = \|\boldsymbol{\lambda}(\tau+1)-\boldsymbol{\lambda}(\tau)\|_2^2.$$

Therefore we have

$$F(\boldsymbol{\lambda}(\tau+1)) \leq F(\boldsymbol{\lambda}(\tau)) - \left(\frac{1}{\gamma_m}-\frac{K}{2}\right)\|\boldsymbol{\lambda}(\tau+1)-\boldsymbol{\lambda}(\tau)\|_2^2.$$

(b) $\boldsymbol{\lambda}(\tau+1)=\boldsymbol{\lambda}(\tau)$ is equivalent to $\mathbf{Z}^j(\tau+1)=\boldsymbol{\lambda}_j(\tau), \forall j$. Moreover, if $\mathbf{Z}^j(\tau+1)=\boldsymbol{\lambda}_j(\tau), \forall j$, then from the definition of each gradient projection, we know it is optimal. Conversely, if $\boldsymbol{\lambda}(\tau)$ minimizes $F(\boldsymbol{\lambda}(\tau))$ over the set Λ, then the gradient projection always projects to the original point, hence $\mathbf{Z}^j(\tau+1)=\boldsymbol{\lambda}_j(\tau), \forall j$. See also [48, Ch 3 Prop. 3.3(b)] for reference.

(c) Since $F(\boldsymbol{\lambda})$ is continuously differentiable, the gradient mapping is continuous. The projection mapping is also continuous. T is the composition of the two and is therefore continuous. ■

Proof of Theorem 6: Lemma 12 is parallel to that of Proposition 3.3 in Ch 3 of [48], and Theorem 6 here is parallel to Proposition 3.4 in Ch 3 of [48]. Therefore, the proof for Proposition 3.4 immediately applies to Theorem 6. We also have $F(\boldsymbol{\lambda})$ is convex in $\boldsymbol{\lambda}$, which completes the proof. ■

E. Proofs for Algorithm 3

We use the following additional lemmas to prove the convergence result of Algorithm 3.

Lemma 13. *Under Algorithm 3, $\boldsymbol{\lambda}(\tau)\in\Lambda'$, $\forall\tau=0,1,2,\ldots$*

Proof: Since $\Lambda\subset\Lambda'$, we know the intial point $\boldsymbol{\lambda}(0)\in\Lambda'$ and the optimal solution $\boldsymbol{\lambda}^*\in\Lambda'$.

If $\boldsymbol{\lambda}(\tau)\in\Lambda'$, then the choice of $\gamma_j^{\downarrow}$ ensures $\lambda_{ij}(\tau+1)\geq 0$. Moreover, the choice of $\theta_j(\tau)$ and the update rule (19) give

$$\sum_{i\in\Omega_j(\tau)}\lambda_{ij}(\tau+1)$$
$$=\sum_{i\in\Omega_j(\tau)}\lambda_{ij}(\tau)-\gamma_j(\tau)\Big(\sum_{i\in\Omega_j(\tau)}(\nabla_{ij}F(\tau)-\theta_j(\tau))$$
$$=\sum_{i\in\Omega_j(\tau)}\lambda_{ij}(\tau).$$

Since $\lambda_{ij}(\tau+1)=\lambda_{ij}(\tau)$ for $i\notin\Omega_j(\tau)$, we have $\sum_i\lambda_{ij}(\tau+1)=\sum_i\lambda_{ij}(\tau)=L_j$.

Finally, the definition of $\gamma_j^{\uparrow}$ ensures

$$\lambda_i(\tau+1)=\sum_{j:i\in\Gamma_j^{\uparrow}(\tau)}\lambda_{ij}(\tau+1)+\sum_{j:i\notin\Gamma_j^{\uparrow}(\tau)}\lambda_{ij}(\tau+1)$$
$$\leq\sum_{j:i\in\Gamma_j^{\uparrow}(\tau)}\Big(\lambda_{ij}(\tau)-\gamma_j^{\uparrow}(\tau)(\nabla_{ij}F(\tau)-\theta_j(\tau))\Big)+\sum_{j:i\notin\Gamma_j^{\uparrow}(\tau)}\lambda_{ij}(\tau)$$
$$\leq\sum_j\lambda_{ij}(\tau)+\frac{\phi+\beta M_i/2}{\phi+\beta M_i}M_i\mu_i-\sum_j\lambda_{ij}(\tau)$$
$$=\frac{\phi+\beta M_i/2}{\phi+\beta M_i}M_i\mu_i$$

■

Lemma 14. *For all $\boldsymbol{\lambda}^a\in\Lambda'$, and all $\boldsymbol{\lambda}^b\in\Lambda'$,*

$$\left\|\nabla F(\boldsymbol{\lambda}^b)-\nabla F(\boldsymbol{\lambda}^a)\right\|_2\leq K'\left\|\boldsymbol{\lambda}^b-\boldsymbol{\lambda}^a\right\|_2.$$

where K' is defined in Algorithm 3.

The proof of this lemma is similar to that of Lemma 11 except that the constraint $\lambda_i\leq\frac{\phi}{\phi+\beta M_i}M_i\mu_i$ is replaced by $\lambda_i\leq\frac{\phi+\beta M_i/2}{\phi+\beta M_i}M_i\mu_i$, resulting in different Lipschitz modulus.

Lemma 15. *Let $\gamma(\tau)=\max_j\gamma_j(\tau)$. Then under Algorithm 3, $F(\boldsymbol{\lambda}(\tau+1))\leq F(\boldsymbol{\lambda}(\tau))-(\frac{1}{\gamma(\tau)}-\frac{K'}{2})\|\boldsymbol{\lambda}(\tau+1)-\boldsymbol{\lambda}(\tau)\|_2^2$.*

Although this result seems similar to a standard one proved by the projection argument, here we do not have a projection. Therefore we devise a different proof technique.

Proof: From Lemma 14 and Proposition A.32 in [48],

$$F(\boldsymbol{\lambda}(\tau+1))\leq F(\boldsymbol{\lambda}(\tau))+(\boldsymbol{\lambda}(\tau+1)-\boldsymbol{\lambda}(\tau))'\nabla F(\tau)+\frac{K'}{2}\|\boldsymbol{\lambda}(\tau+1)-\boldsymbol{\lambda}(\tau)\|_2^2,$$

where we take $\boldsymbol{\lambda}(\tau)$ as a $|N||J|$-dimension vector. The proof is completed by expanding the second term as

$$(\boldsymbol{\lambda}(\tau+1)-\boldsymbol{\lambda}(\tau))'\nabla F(\tau)$$
$$=\sum_j\sum_{i\in\Omega_j(\tau)}(-\gamma_j(\tau)(\nabla_{ij}F(\tau)-\theta_j(\tau))\nabla_{ij}F(\tau)$$
$$=-\sum_j\gamma_j(\tau)\sum_{i\in\Omega_j(\tau)}(\nabla_{ij}F(\tau)-\theta_j(\tau))(\nabla_{ij}F(\tau)-\theta_j(\tau))$$
$$=-\sum_j\frac{1}{\gamma_j(\tau)}(\lambda_{ij}(\tau+1)-\lambda_{ij}(\tau))^2$$
$$\leq-\frac{1}{\gamma(\tau)}\|\lambda(\tau+1)-\lambda(\tau)\|_2^2,$$

where the second step uses the definition in (17). ■

With the lemmas above, we now prove Theorem 7.

Proof of Theorem 7: Let $\mathcal{J}^\epsilon\equiv\{(i,j):0<\lambda_{ij}\leq\epsilon$ and $\nabla_{ij}F(\boldsymbol{\lambda}(\tau))>\theta_j(\tau)\}$ be those loads prevented from decreasing in (19). We first show that Algorithm 3 has an accumulation point $\boldsymbol{\lambda}^a$ satisfying the KKT conditions except for (22) for $(i,j)\in\mathcal{J}^\epsilon$. We then construct an optimization GLB′ solved by $\boldsymbol{\lambda}^a$ whose KKT conditions match GLB-Q except for (22) for $(i,j)\in\mathcal{J}^\epsilon$, and bound the difference between its optimum and that of GLB-Q.

(1) Note $\gamma_j^{\downarrow}(\tau)$ is bounded below since $\lambda_{ij}(\tau)>\epsilon$ for any $i\in\Gamma_j^{\downarrow}(\tau)$ and $\nabla_{ij}F(\boldsymbol{\lambda}(\tau))-\theta_j(\tau)$ is bounded above; $\gamma_j^{\uparrow}(\tau)$ is also bounded below since $\frac{\phi+\beta M_i/2}{\phi+\beta M_i}M_i\mu_i-\lambda_i(\tau)\geq\frac{\beta M_i/2}{\phi+\beta M_i}M_i\mu_i$ and $\nabla_{ij}F(\boldsymbol{\lambda}(\tau))-\theta_j(\tau)$ is bounded above. Since the third case in (18) is constant, $\gamma_j(\tau)$ is bounded below. Hence $\|\boldsymbol{\lambda}(\tau+1)-\boldsymbol{\lambda}(\tau)\|_2^2$ converges to 0 only if the corresponding KKT conditions hold *except* for the complementary slackness conditions in (22) for the $(i,j)\in\mathcal{J}^\epsilon$. Since there is an $\varepsilon>0$ such that $\gamma(\tau)<2/K'-\varepsilon$, Lemma 15 ensures Algorithm 3 makes a substantial decrease each step until the KKT conditions hold except for (22) for $(i,j)\in\mathcal{J}^\epsilon$.

(2) Algorithm 3 has an accumulation point, $\boldsymbol{\lambda}^a$, since $F(\boldsymbol{\lambda})$ converges due to being bounded below, and $\boldsymbol{\lambda}$ comes from a compact set. Next, we construct GLB′ solved by $\boldsymbol{\lambda}^a$ whose KKT conditions match those of GLB-Q except for (22) for $(i,j)\in\mathcal{J}^\epsilon$, and show that $|F(\boldsymbol{\lambda}^a)-F(\boldsymbol{\lambda}^*)|=O(\epsilon)$, where ϵ is the error tolerance in λ_{ij}.

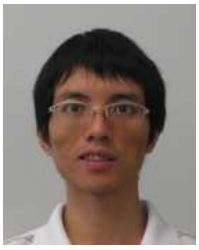
Let $\boldsymbol{\lambda}^\epsilon$ be the matrix with $\lambda^\epsilon_{ij} = \lambda^a_{ij}$, if $(i,j) \in \mathcal{J}^\epsilon$, and $\lambda^\epsilon_{ij} = 0$ otherwise. Define $\lambda^\epsilon_i = \sum_j \lambda^\epsilon_{ij}$ and denote by $\boldsymbol{\lambda}^\epsilon_i$ the vector of $(\lambda^\epsilon_{ij})_{j\in J}$. Define GLB$'$ to be solving (5a) subject to (5b), (3b), (3d) and $\lambda_{ij} \geq \lambda^\epsilon_{ij}$ for all $i \in N$ and $j \in J$. The KKT conditions of GLB$'$ match those of GLB-Q, except that the analog of (22) is $\delta_{ij}(\lambda_{ij} - \lambda^\epsilon_{ij}) = 0$. For $(i,j) \in \mathcal{J}^\epsilon$, this holds by the definition of λ^ϵ. All other conditions were established in step 1) above, and so $\boldsymbol{\lambda}^a$ optimizes GLB$'$.

If $\boldsymbol{\lambda}^* \geq \boldsymbol{\lambda}^\epsilon$, $\boldsymbol{\lambda}^a$ optimizes GLB-Q, and the result is proved. Otherwise, we perturb $\boldsymbol{\lambda}^*$ to yield $\boldsymbol{\lambda}''$ which is feasible for GLB$'$, and bound the resulting increase in cost, as follows.

First construct a solution $\boldsymbol{\lambda}'$ from $\boldsymbol{\lambda}^*$ where $\lambda'_i \geq \lambda^\epsilon_i$. If there exist some $i \in S^\epsilon$ with $\lambda^*_i < \lambda^\epsilon_i$, we construct the new $\boldsymbol{\lambda}'_i$ by moving some traffic $\lambda^\epsilon_i - \lambda^*_i$ from $i \notin S^\epsilon$ to these data centers $i \in S^\epsilon$ to make $\lambda'_i = \lambda^\epsilon_i$. Now we compare $F(\boldsymbol{\lambda}')$ and $F(\boldsymbol{\lambda}^*)$. By moving $\lambda^\epsilon_i - \lambda^*_i$, we decrease the cost on some $i \notin S^\epsilon$, but increase that on $i \in S^\epsilon$. Since $\lambda^\epsilon_i - \lambda^*_i \leq \lambda^\epsilon_i \leq |J|\epsilon$ and $\epsilon \leq \min_i \left(M_i\mu_i/(1+\sqrt{\beta/p_i})\right)/|J|$, $\lambda'_i = \lambda^\epsilon_i \leq |J|\epsilon \leq M_i\mu_i/(1+\sqrt{\beta/p_i})$ for $i \in S^\epsilon$. Within this region, $m'_i = (1+\sqrt{\beta/p_i})\lambda'_i/\mu_i$ optimizes (5a). Neglecting delay d_{ij}, the increase in term i is no larger than $\beta|J|\epsilon\left(1+\sqrt{p_i/\beta}\right)^2/\mu_i$. The delay cost increase is at most $\beta|J|\epsilon\max_j d_{ij}$. Thus $F(\boldsymbol{\lambda}') \leq F(\boldsymbol{\lambda}^*) + \beta|J|\epsilon\sum_i\left((1+\sqrt{p_i/\beta})^2/\mu_i + \max_j d_{ij}\right)$.

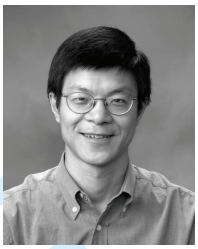
From $\boldsymbol{\lambda}'$ we construct $\boldsymbol{\lambda}''$ by reassigning traffic cyclically to make $\lambda''_{ij} \geq \lambda^\epsilon_{ij}, \forall i,j$. The total cost increase is bounded by $\beta|J|\epsilon\sum_i \max_j d_{ij}$. Therefore $F(\boldsymbol{\lambda}'') \leq F(\boldsymbol{\lambda}^*) + \beta|J|\epsilon\sum_i\left((1+\sqrt{p_i/\beta})^2/\mu_i + 2\max_j d_{ij}\right) = F(\boldsymbol{\lambda}^*) + B\epsilon$. To complete the proof, note $F(\boldsymbol{\lambda}^a) \leq F(\boldsymbol{\lambda}'')$, since $\boldsymbol{\lambda}''$ is feasible for GLB$'$. ■

Zhenhua Liu received the B.E. degree in measurement and control, M.S. degree in computer science and technology (both with honor) from Tsinghua University, Beijing, China, in 2006 and 2009, respectively. He is current a Computer Science PhD student at the California Institute of Technology (Caltech), Pasadena, CA. His current research interests include sustainable computing and networking systems, renewable energy integration, cloud-based platform for big data and energy management, optimization, game theory, online and distributed algorithms. He received the "Best Student Paper" award in ACM GreenMetrics'11, "Best Paper" award in IGCC'12, and the Pick of the Month award by IEEE STC on Sustainable Computing. He was a research associate (intern) with HP Labs, Palo Alto, in 2011-2013, and his work on the industry's first Net-zero Energy Data Center was named a 2013 Computerworld Honors Laureate.



Minghong Lin received the B.Sc. degree in Computer Science from University of Science and Technology of China in 2006, and the M.Phil degree in Computer Science from the Chinese University of Hong Kong in 2008. Currently he is a Ph.D. student in Computer Science at California Institute of Technology. His research interests include energy efficient computing, online algorithms and nonlinear optimization. He has received best paper award at IEEE INFOCOM'11, IGCC'12 and best student paper award at ACM GREENMETRICS'11.

Adam Wierman is a Professor in the Department of Computing and Mathematical Sciences at the California Institute of Technology, where he is a member of the Rigorous Systems Research Group (RSRG). He received his Ph.D., M.Sc. and B.Sc. in Computer Science from Carnegie Mellon University in 2007, 2004, and 2001, respectively. He received the ACM SIGMETRICS Rising Star award in 2011, and has also received best paper awards at ACM SIGMETRICS, IFIP Performance, IEEE INFOCOM, and ACM GREENMETRICS. He has also received multiple teaching awards, including the Associated Students of the California Institute of Technology (ASCIT) Teaching Award. His research interests center around resource allocation and scheduling decisions in computer systems and services. More specifically, his work focuses both on developing analytic techniques in stochastic modeling, queueing theory, scheduling theory, and game theory, and applying these techniques to application domains such as energy-efficient computing, data centers, social networks, and the electricity grid.



Steven Low (F'08) Steven Low (F08) received the B.S. degree from Cornell University, Ithaca, NY, USA, and the Ph.D. degree from the University of California, Berkeley, USA, both in electrical engineering. He is a Professor of the Computing and Mathematical Sciences and Electrical Engineering Departments at the California Institute of Technology, Pasadena, CA, USA. Before that, he was with AT&T Bell Laboratories, Murray Hill, NJ, USA, and the University of Melbourne, Australia. He is a Senior Editor of the IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, a Senior Editor of the IEEE TRANSACTIONS ON CONTROL OF NETWORK SYSTEMS, a Steering Committee Member of the IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING, and on the editorial board of NOW Foundations and Trends in Networking, and in Power Systems. He also served on the editorial boards of IEEE/ACM TRANSACTIONS ON NETWORKING, IEEE TRANSACTIONS ON AUTOMATIC CONTROL, ACM Computing Surveys, Computer Networks Journal.

Lachlan Andrew (M'97-SM'05) received the B.Sc., B.E. and Ph.D. degrees in 1992, 1993, and 1997, from the University of Melbourne, Australia. Since 2008, he has been an associate professor at Swinburne University of Technology, Australia, and since 2010 he has been an ARC Future Fellow. From 2005 to 2008, he was a senior research engineer in the Department of Computer Science at Caltech. Prior to that, he was a senior research fellow at the University of Melbourne and a lecturer at RMIT, Australia. His research interests include energy-efficient networking and performance analysis of resource allocation algorithms. He was co-recipient of the best paper award at IGCC2012, IEEE INFOCOM 2011 and IEEE MASS 2007. He is a member of the ACM.