Greening Geographical Load Balancing

Thesis by

Zhenhua Liu

In Partial Fulfillment of the Requirements
for the Degree of
Master of Science

California Institute of Technology
Pasadena, California

2011
(Submitted May 31, 2011)

Acknowledgements

I feel extremely fortunate to be co-advised by Prof. Steven Low and Prof. Adam Wierman. I would like to express my utmost gratitude to them, for their enormous guidance and help. Their patience and encouragement accompanied me to hurdle all the obstacles during the past two years. I really learned a lot from both of them.

I am also grateful to my collaborators, Lachlan Andrew and Minghong Lin. They have always provided insightful discussions and constructive suggestions. The atmosphere and environment of the RSRG, Computer Science department and Caltech are amazing and helpful. It is really a pleasure to work and study here.

Last but not least, my family provided me a pleasant environment, which helped a lot during this research. I would like to thank my wife Zheng especially for her understanding and love all the time. This thesis would not have been possible without their support!

Abstract

Energy expenditure has become a significant fraction of data center operating costs. Recently, "geographical load balancing" has been suggested to reduce energy cost by exploiting the electricity price differences across regions. However, this reduction of cost can paradoxically increase total energy use. This work explores whether the geographical diversity of internet-scale systems can additionally be used to provide environmental gains.

We first focus on geographical load balancing, which is modeled as a convex optimization problem. We derive two distributed algorithms for achieving optimal geographical load balancing and characterize the optimal solutions.

Then we continue to use the framework and algorithms to investigate whether geographical load balancing can encourage use of "green" renewable energy and reduce use of "brown" fossil fuel energy. Here we consider two approaches, namely, dynamic pricing and local renewables.

For the dynamic pricing case, our numeric results show that if electricity is dynamically priced in proportion to the instantaneous fraction of the total energy that is brown, then geographical load balancing provides significant reductions in brown energy use. However, the benefits depend strongly on the degree to which systems accept dynamic energy pricing and the form of pricing used.

For the local renewables case, we perform a trace-based study to evaluate three issues related to achieving this goal: the impact of geographical load balancing, the role of storage, and the optimal mix of renewables. Our results highlight that geographical load balancing can significantly reduce the required capacity of renewable energy by using the energy more efficiently with "follow the renewables" routing. Further, our results show that small-scale storage can be useful, especially in combination with geographical load balancing, and that an optimal mix of renewables includes significantly more wind than photovoltaic solar.

Contents

Acknowledgements **iii**

Abstract **iv**

1 Introduction **1**

2 Geographical Load Balancing **5**

2.1 Model and Notation 5

2.1.1 The workload model 5

2.1.2 The data center cost model 5

2.1.3 The geographical load balancing problem 7

2.1.4 Practical considerations 9

2.2 Characterizing the optima 9

2.2.1 Uniqueness of optimal solution 9

2.2.2 Sparsity of routing 10

2.3 Algorithms 11

2.3.1 Algorithm 1: Gauss-Seidel iteration 11

2.3.2 Algorithm 2: Distributed gradient projection 13

2.4 Case study 15

2.4.1 Experimental setup 15

2.4.2 Performance evaluation 17

3 Exploring Renewables **21**

3.1 Dynamic pricing 21

3.1.1 Experimental setup 21

3.1.2 The importance of dynamic pricing 24

3.2 Local renewables 25

3.2.1 Experimental setup 25

3.2.2 Results 30

4 Concluding remarks **34**

Bibliography **35**

Appendices **40**

A Proofs for Chapter 2.2 **40**

A.1 Optimality conditions 40

A.2 Characterizing the optima 42

B Proofs for Chapter 2.3 **45**

B.1 Algorithm 1 45

B.2 Algorithm 2 46

Chapter 1

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 [45].

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, multi-core servers, DC power, etc. have led to significant improvements in the Power Usage Effectiveness (PUE) of data centers; see [8, 51, 26, 28]. 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.

In recent years, many papers have illustrated the potential of geographical load balancing to provide significant cost savings for data centers, e.g., [32, 42, 45, 46, 48, 53] and the references therein. The goal of the current thesis is different. Our goal is to explore the social impact of geographical load balancing systems. In particular, geographical load balancing aims to reduce energy costs, but this can come at the expense of increased total energy usage: by routing to a data center farther from the request source to use cheaper energy, the data center may need to complete the job faster, and so use more service capacity, and thus energy, than if the request was served closer to the source.

In contrast to this negative consequence, geographical load balancing also 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, unpredictable 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), renewable sources pose a significant challenge. A key technique for handling the unpredictability of renewable sources is *demand-response*, which entails the grid adjusting the demand by changing the electricity price [3]. However, demand response entails a *local* customer curtailing use. In contrast, the demand of internet-scale systems is flexible geographically; thus traffic can be routed to different regions to "follow the renewables", providing demand-response without service interruption. Since data centers represent a significant and growing fraction of total electricity consumption, and the IT infrastructure is already in place, geographical load balancing has the potential to provide an extremely 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 fixed price contracts that are now typical toward some degree of dynamic pricing, with lower prices when green energy is available. The demand response markets currently in place provide a natural way for this transition to occur, and there is already evidence of some data centers participating in such markets [3].

On the other hand, data centers are often powered by a "green" portfolio of energy already [30, 35, 37]. However, most studies of powering data centers *entirely* with renewable energy have focused on powering individual data centers, e.g., [19, 20]. These have shown that it is challenging to power a data center using only local wind and solar energy without large-scale storage, due to the intermittency and unpredictability of these sources.

Our goal here is to illustrate that the geographical diversity of internet-scale services significantly improves the efficiency of the usage of renewable energy. This numerical study complements testbeds such as [40]. Further, we illustrate that algorithmic solutions can play a vital role in reducing the necessary capacity of renewable energy installed.

Specifically, we perform numerical experiments using real traffic workloads and real data about the availability of renewables combined with an analytic model for a geographically distributed system. Using this setup, we investigate issues related to the feasibility of powering an internet-scale system (nearly) completely with renewable energy.

We have three major contributions in this work. (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. (3) We study the case where data center is (nearly) completely powered by local renewables.

Contribution (1): To derive distributed geographical load balancing algorithms we use a simple but general model, described in detail in Chapter 2.1. 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 define both how requests should be routed to data centers and how to allocate capacity in each data center (i.e., how many servers are kept in active/energy-saving states).

In Chapter 2.2, we characterize the optimal geographical load balancing solutions and show that they have practically appealing properties, such as sparse routing tables. Then, in Chapter 2.3, we use the previous characterization to give two distributed algorithms which provably compute the optimal routing and provisioning decisions, and which require different types of coordination of computation. Finally, we evaluate the distributed algorithms in a trace-driven numeric simulation of a realistic, distributed, internet-scale system (Chapter 2.4). The results show that a cost saving of over 40% during light-traffic periods is possible.

Contribution (2): In Chapter 3.1 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 models for the availability of wind and solar energy over time.

When the data center incentive is aligned with the social objective or 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 3.1), causing significant social benefit. In contrast, we also determine the wasted brown energy when prices are static, or are dynamic but do not align data center and social objectives.

Contribution (3): Our study in Chapter 3.2 yields three key insights.

First, GLB significantly reduces the capacity of renewables needed to move toward a "green" system, since GLB allows the system to use "follow the renewables" routing, which reduces both the financial cost and the "brown" non-renewable energy usage. However, we also show the importance of using a fast control time-scale for GLB. If routing and capacity decisions are made only once an hour, then significantly more brown energy is consumed than if the adjustments are made every ten minutes. Unfortunately, adjustments at this faster time-scale may not be feasible due to server wear-and-tear and other concerns.

Second, we investigate the value of storage when using renewable energy. Often large-scale storage is viewed as essential for moving toward a completely renewable energy portfolio. However, our study shows that small-scale storage in combination with GLB is sufficient in moving to a portfolio of nearly completely renewable energy sources. This is particularly exciting since the UPSs in use at data centers today could be used to provide small-scale storage with few engineering changes.

Third, we find that wind is more valuable than solar for internet-scale systems, especially when GLB is used, because wind has little correlation across locations, and is available during both night and day. Thus, if one aggregates over many locations, there is much less variation in the availability [7]. The optimal mix seems to be dominated by wind, but include some solar to handle the peak workload around noon. For the traces we consider, the optimal portfolio is 80% wind and 20% solar. This ratio may depend on the quality of the local wind resources; moreover, workloads with higher (lower) diurnal peak-to-mean ratios may benefit from a higher (lower) solar component.

Chapter 2

Geographical Load Balancing

In this section, we introduce the geographical load balancing problem. We first model the problem as a convex optimization problem, then characterize the optimal solutions and use the optimal properties to design two distributed algorithms. We also provide numeric results for the performance evaluation.

2.1 Model and Notation

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

2.1.1 The workload model

We consider a discrete-time model whose timeslot matches the timescale at which routing decisions and capacity provisioning decisions can be updated. There is a (possibly long) interval of interest $t \in \{1, \dots, T\}$. There are $|J|$ geographically concentrated sources of requests, i.e., "cities", and the mean arrival rate from source j at time t is $L_j(t)$. Job interarrival times are assumed to be much shorter than a timeslot, so that provisioning can be based on the average arrival rate during a slot. In practice, T could be a month and a slot length could be 1 hour. Our analytic results make no assumptions on $L_j(t)$; however to provide realistic estimates we use real-world traces to define $L_j(t)$ in Chapters 2.4 and 3.

2.1.2 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: (i) determining $\lambda_{ij}(t)$, the amount of traffic routed from source j to data center i; and (ii) determining $m_i(t) \in \{0, \dots, M_i\}$, the number of active servers at data center 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. Algorithms for these decisions are the focus of Chapter 2.3.

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 [32] provides a natural way to incorporate such costs if desired.

Energy cost. To capture the geographic 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. 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 convex in m_i. This formulation is quite general, and captures, for example, the common charging plan of a fixed price per kWh plus an additional "demand charge" for the peak of the average power used over a sliding 15 minute window [41]. Additionally, it can capture a wide range of models for server power consumption, e.g., energy costs as an affine function of the load, see [16], or as a polynomial function of the speed, see [54, 6].

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

$$\mathcal{E}_i(t) = g_i(t, m_i(t), \lambda_i(t)). \tag{2.1}$$

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

To model the delay, we consider its two components: the network delay experienced while the request is outside of the data center and the queueing delay experienced while the request is at the data center.

To model the *network delay*, we let $d_{ij}(t)$ denote the network delay experienced by a request from source j to data center i during timeslot t. We make no requirements on the structure of the $d_{ij}(t)$.

To model the *queueing delay*, we let $f_i(m_i, \lambda_i)$ denote the queueing delay at data center i given m_i active servers and an arrival 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

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

$f_i(m_i, \lambda_i) = \infty$ for $\lambda_i \geq m_i\mu_i$. Elsewhere, we assume f_i is finite, continuous and differentiable. Note that these assumptions are satisfied by most standard queueing formulae, e.g., the mean 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:

$$\mathcal{D}_i(t) = \sum\nolimits_{j\in J} \lambda_{ij}(t) r\left(f_i(m_i(t), \lambda_i(t)) + d_{ij}(t)\right). \tag{2.2}$$

2.1.3 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 $[1, T]$. This is captured by the following optimization problem:

$$\begin{aligned}
&\min_{\mathbf{m}(t),\boldsymbol{\lambda}(t)} \sum\nolimits_{t=1}^{T} \sum\nolimits_{i\in N} \left(\mathcal{E}_i(t) + \mathcal{D}_i(t)\right) && && \text{(2.3a)}\\
&\text{s.t.} \sum\nolimits_{i\in N} \lambda_{ij}(t) = L_j(t), && \forall j \in J && \text{(2.3b)}\\
&\lambda_{ij}(t) \geq 0, && \forall i \in N, \forall j \in J && \text{(2.3c)}\\
&0 \leq m_i(t) \leq M_i, && \forall i \in N && \text{(2.3d)}\\
&m_i(t) \in \mathbb{N}, && \forall i \in N && \text{(2.3e)}
\end{aligned}$$

To simplify (2.3), note that Internet data centers typically contain thousands of active servers. So, we can relax the integer constraint in (2.3) and round the resulting solution with minimal increase in cost. Also, because this model neglects the cost of turning servers on or off, the optimization decouples into independent sub-problems for each timeslot t. For the analysis *we consider only a single interval and omit the explicit time dependence.*[2] Thus (2.3) becomes

$$\begin{aligned}
&\min_{\mathbf{m},\boldsymbol{\lambda}} \sum_{i\in N} g_i(m_i, \lambda_i) + \sum_{i\in N}\sum_{j\in J} \lambda_{ij} r(d_{ij} + f_i(m_i, \lambda_i)) && && \text{(2.4a)}\\
&\text{s.t.} \sum\nolimits_{i\in N} \lambda_{ij} = L_j, && \forall j \in J && \text{(2.4b)}\\
&\lambda_{ij} \geq 0, && \forall i \in N, \forall j \in J && \text{(2.4c)}\\
&0 \leq m_i \leq M_i, && \forall i \in N, && \text{(2.4d)}
\end{aligned}$$

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. However, a distributed solution algorithm is usually required, such as

[2]Time-dependence of L_j and prices is re-introduced for, and central to, the numeric results in Chapters 2.4 and 3.1.

those derived in Chapter 2.3.

In contrast to prior work studying geographical load balancing, it is important to observe that this paper is the first, to our knowledge, to incorporate jointly optimizing the total energy cost and the end-to-end user delay with consideration of both price diversity and network delay diversity.

GLB provides a general framework for studying geographical load balancing. However, the model 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 [50] 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. This model uses a lost revenue function $r(d) = \beta d$, for constant β. Though it is difficult to choose a "universal" form for the lost revenue associated with delay, there is evidence that it is linear within the range of interest for sites such as Google, Bing, and Shopzilla [15]. GLB then simplifies to

$$\min_{\mathbf{m},\boldsymbol{\lambda}} \sum_{i\in N} g_i(m_i,\lambda_i) + \beta \left(\sum_{i\in N} \lambda_i f_i(m_i,\lambda_i) + \sum_{i\in N}\sum_{j\in J} d_{ij}\lambda_{ij} \right) \tag{2.5}$$

subject to (3.4a)–(3.4c). We call this optimization GLB-LIN.

Queueing-based delay. We occasionally specify the form of f and g using queueing models. This provides increased intuition 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, each with arrival rate λ_i/m_i. Moreover, if each queue is an M/GI/1 Processor Sharing (PS) queue, then $f_i(m_i, \lambda_i) = 1/(\mu_i - \lambda_i/m_i)$. 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 p_i regardless of λ_i.

Under these restrictions, the GLB formulation becomes:

$$\min_{\mathbf{m},\boldsymbol{\lambda}} \sum_{i\in N} p_i m_i + \beta \sum_{j\in J}\sum_{i\in N} \lambda_{ij} \left(\frac{1}{\mu_i - \lambda_i/m_i} + d_{ij} \right) \tag{2.6a}$$

subject to (3.4a)–(3.4c) and the additional constraint

$$\lambda_i \leq m_i \mu_i \quad \forall i \in N. \tag{2.6b}$$

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. Similarly $\mathbf{m} = (m_i)_{i\in N}$ and $\boldsymbol{\lambda} = (\lambda_{ij})_{i\in N, j\in J}$.

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 $\partial F_i(m_i, \lambda_i)/\partial m_i = 0$.

2.1.4 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., [5, 13, 18, 52]. 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., [14, 34, 44], 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 [43]. Within the GLB model, we have implicitly assumed that there exists a proxy/DNS server co-located with each source.

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., [49], and mapping and synthetic coordinate approaches, e.g., [29, 39]. We discuss the sensitivity of our algorithms to error in these estimates in Chapter 2.4.

2.2 Characterizing the optima

We now provide characterizations of the optimal solutions to GLB, which are important for proving convergence of the distributed algorithms of Chapter 2.3. 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.

2.2.1 Uniqueness of optimal solution

To begin, note that GLB has at least one optimal solution. This can be seen by applying Weierstrass theorem [9], 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. 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.*

2.2.2 Sparsity of routing

It would be impractical if the optimal solutions to GLB required that traffic from each source was divided up among (nearly) all of the data centers. In general, each λ_{ij} could be non-zero, yielding $|N| \times |J|$ flows of traffic 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:

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 an optimal solution into a sparse optimal solution.

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 a 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 traffic 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 $m_i < M_i$.*

2.3 Algorithms

We now focus on GLB-Q and present two distributed algorithms that solve it, and prove their convergence.

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. However there is a strong case for Internet-scale systems to outsource route selection [53]. 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 hole 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 both energy price and network propagation delay information. The most closely related work [46] 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 an $M/M/1$ queue) but neglects the end-to-end user delay. Conversely, [53] uses a simple, efficient algorithm to coordinate the "replica-selection" decisions, but assumes the capacity at each data center is fixed. Other related works, e.g., [46, 48, 42], either do not provide provable guarantees or ignore diverse network delays and/or prices.

2.3.1 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 to each other. 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:

$$\min_{\boldsymbol{\lambda}_j \in \Lambda_j} \min_{m_i \in \mathcal{M}_i} \sum_{i \in N} \left(p_i m_i + \frac{\beta \lambda_i}{\mu_i - \lambda_i / m_i} \right) + \beta \sum_{i,j} \lambda_{ij} d_{ij} \tag{2.7}$$

$$\mathcal{M}_i := [0, M_i] \qquad \Lambda_j := \{\boldsymbol{\lambda}_j \mid \boldsymbol{\lambda}_j \geq 0, \sum_{i \in N} \lambda_{ij} = L_j\} \tag{2.8}$$

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 (2.7) over their own variables m_i:

$$\min_{m_i \in \mathcal{M}_i} \left(p_i m_i + \frac{\beta \lambda_i(\tau)}{\mu_i - \lambda_i(\tau)/m_i} \right) \tag{2.9}$$

In phase j of iteration τ, proxy j minimizes (2.7) 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:

$$\lambda_i(\tau, j) := \sum_{l<j} \lambda_{il}(\tau+1) + \sum_{l>j} \lambda_{il}(\tau) \tag{2.10}$$

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. When there are fewer data centers than proxies, this has less overhead than direct messaging.

In summary, the algorithm is as follows (noting that the minimization (2.9) 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*

$$m_i(\tau+1) := \left[\left(1 + \frac{1}{\sqrt{p_i/\beta}} \right) \cdot \frac{\lambda_i(\tau)}{\mu_i} \right]^{M_i} \tag{2.11}$$

$$\boldsymbol{\lambda}_j(\tau+1) := \arg\min_{\boldsymbol{\lambda}_j \in \Lambda_j} \sum_{i \in N} \frac{\lambda_i(\tau, j) + \lambda_{ij}}{\mu_i - (\lambda_i(\tau, j) + \lambda_{ij})/m_i(\tau+1)} + \sum\nolimits_{i \in N} \lambda_{ij} d_{ij}. \tag{2.12}$$

Since GLB-Q generally has multiple optimal $\boldsymbol{\lambda}_j^*$, Algorithm 1 is not guaranteed to converge to one 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 B.

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 Chapter 2.4.

2.3.2 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 (2.12) 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

$$\min_{\boldsymbol{\lambda}_j \in \Lambda_j} \sum_{j\in J} F_j(\boldsymbol{\lambda}) \tag{2.13}$$

where $F(\boldsymbol{\lambda})$ is the result of minimization of (2.7) 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 by (cf. (2.11)):

$$m_i(\lambda_i) := \left[\left(1 + \frac{1}{\sqrt{p_i/\beta}} \right) \cdot \frac{\lambda_i}{\mu_i} \right]^{M_i} \tag{2.14}$$

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 $\prod_j \Lambda_j$. The standard gradient projection algorithm will converge if $\nabla F(\boldsymbol{\lambda})$ is Lipschitz over our feasible set $\prod_j \Lambda_j$. 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

$$\Lambda := \Lambda(\phi) := \prod_j \Lambda_j \ \cap \ \left\{ \boldsymbol{\lambda} \,\middle|\, \lambda_i \le \frac{\phi M_i \mu_i}{\phi + \beta M_i}, \ \forall i \right\}. \tag{2.15}$$

Even though the Λ defined in (2.15) indeed has the desired properties (see Appendix B), 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 the measured arrival rates $\lambda_i(\tau)$ it is told by data centers i, as done in Algorithm 1 (cf. (2.10) 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.[3] Moreover, if $\boldsymbol{\lambda}(0) \in \Lambda$ then $\boldsymbol{\lambda}(\tau) \in \Lambda$ for all iterations τ. In summary, Algorithm 2 is as follows.

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:*

$$\mathbf{z}_j(\tau+1) := \left[\boldsymbol{\lambda}_j(\tau) - \gamma_j \left(\nabla F_j(\boldsymbol{\lambda}(\tau))\right)\right]_{\hat{\Lambda}_j(\tau)} \tag{2.16}$$

$$\boldsymbol{\lambda}_j(\tau+1) := \frac{|J|-1}{|J|}\boldsymbol{\lambda}_j(\tau) + \frac{1}{|J|}\mathbf{z}_j(\tau+1) \tag{2.17}$$

where $\gamma_j > 0$ is a stepsize and $\nabla F_j(\boldsymbol{\lambda}(\tau))$ is given by

$$\frac{\partial F(\boldsymbol{\lambda}(\tau))}{\partial \lambda_{ij}} = \beta \left(d_{ij} + \frac{\mu_i}{\left(\mu_i - \lambda_i(\tau)/m_i(\lambda_i(\tau))\right)^2} \right).$$

Implicit in the description is the requirement that all data centers i compute $m_i(\lambda_i(\tau))$ according to (2.14) 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 $\boldsymbol{\lambda}_j(\tau + 1)$.

[3]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 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 B. 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 Chapter 2.4.

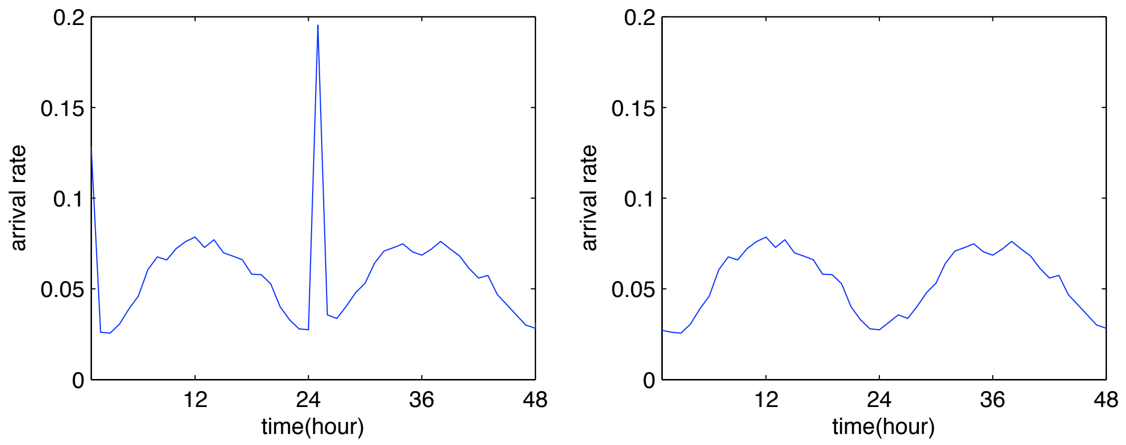
2.4 Case study

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 Chapter 3.1 considers the social perspective (i.e., brown energy usage).

2.4.1 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.

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 2.1(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



(a) Original trace (b) Adjusted trace

Figure 2.1: Hotmail trace used in numerical results.

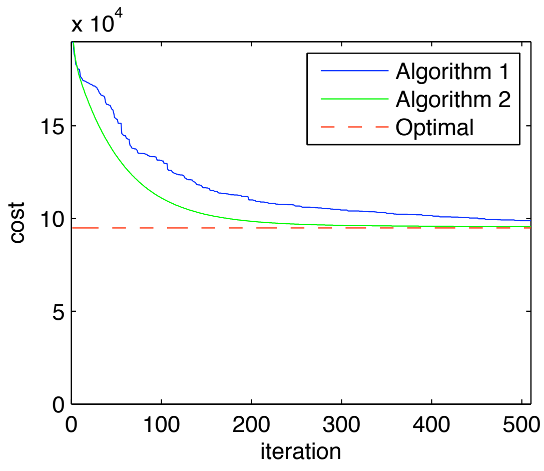
with the spike removed (as illustrated in Figure 2.1(b)) because this leads to a more conservative estimate of the cost savings.

Building on this trace, we construct our workload by placing a source at the geographical center of each mainland US state, co-located with a proxy or DNS server (as described in Chapter 2.1.4). 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 [1].

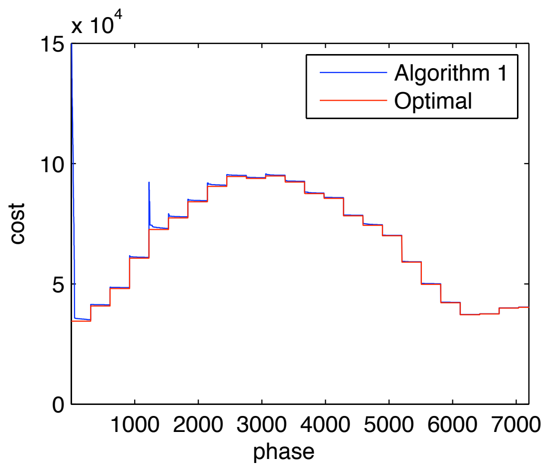
Data center description To model an Internet-scale system, we have 14 data centers, one at the geographic center of each state known to have Google data centers [24]: California, Washington, Oregon, Illinois, Georgia, Virginia, Texas, Florida, North Carolina, and South Carolina.

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 distances between the centers of the two states and comparable to queueing delays. This lower bound on the network delay ignores delay due to congestion or indirect routes.

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 [25]. 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 $\beta = 1$; however Figure 2.3 illustrates the impact of varying β.



(a) Static setting



(b) Dynamic setting

Figure 2.2: Convergence of Algorithm 1 and 2.

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 [46]. 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 Algorithms 1 and 2 compared to those such as [46, 48] 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 [4].

2.4.2 Performance evaluation

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

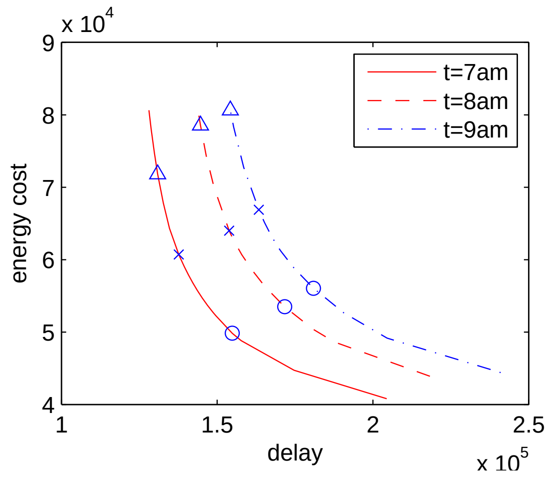


Figure 2.3: Pareto frontier of the GLB-Q formulation as a function of β for three different times (and thus arrival rates), PDT. Circles, x-marks, and triangles correspond to $\beta = 0.4$, 1, and 2.5, respectively.

Convergence We start by considering the convergence of each of the distributed algorithms. Figure 2.2(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 used 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.

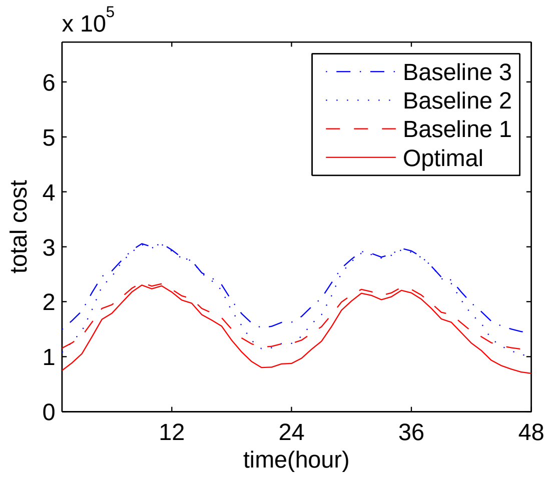
To demonstrate the convergence in a dynamic setting, Figure 2.2(b) shows Algorithm 1's response to the first day of the Hotmail trace, with loads averaged over one-hour intervals for brevity. One iteration (51 phases) is performed every 10 minutes. This figure shows that even the slower algorithm, Algorithm 1, converges fast enough to provide near-optimal cost. Hence, the remaining plots show only the optimal solution.

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 2.3 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'.

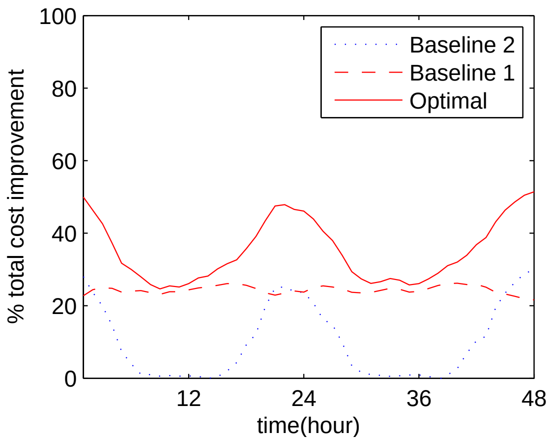
Cost savings To evaluate the cost savings of geographical load balancing, Figure 2.4 compares the optimal costs to those incurred under the three baseline strategies described in the experimental setup. The overall cost, shown in Figures 2.4(a) and 2.4(b), is significantly lower under the optimal solution 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 [46, 48].

To understand where the benefits are coming from, let us consider separately the two components of cost: delay and energy. Figures 2.4(c) and 2.4(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.

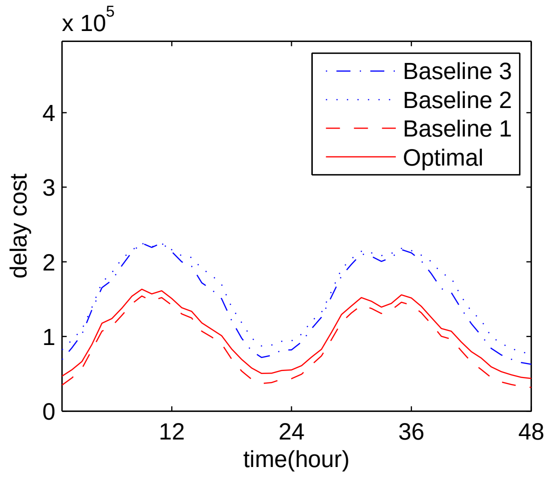
Sensitivity analysis Given that the algorithms all rely on estimates of the L_j and d_{ij} it is important to perform a sensitivity analysis to understand the impact of errors in these parameters on the achieved cost. We have performed such a sensitivity analysis but omit the details for brevity. The results show that even when the algorithms have very poor estimates of d_{ij} and L_j there is little effect on cost. Baseline 2 can be thought of as applying the optimal algorithm to very poor estimates of d_{ij} (namely $d_{ij} = 0$), and so the Figure 2.4(a) provides some illustration of the effect of estimation error.



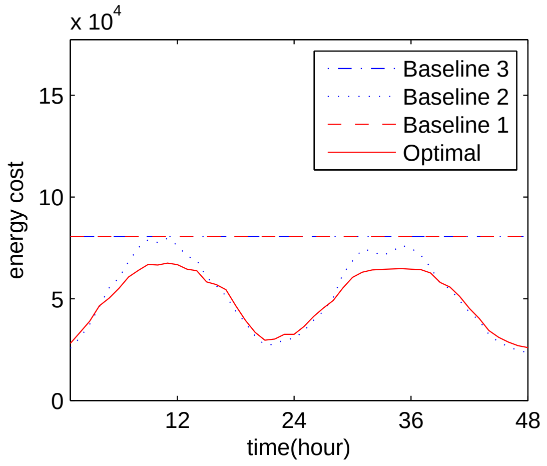
(a) Total cost



(b) Total cost (% improvement vs. Baseline 3)



(c) Delay cost



(d) Energy cost

Figure 2.4: Impact of information used on the cost incurred by geographical load balancing.

Chapter 3

Exploring Renewables

We now shift focus from the cost savings of the data center operator to the social impact of geographical load balancing. We first 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 pricing. Then we investigate the case of local renewable supply.

3.1 Dynamic 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 markets). The experiments in this section show that if dynamic pricing is done optimally, then geographical load balancing can provide significant social benefits.

3.1.1 Experimental setup

To explore the social impact of geographical load balancing, we use the setup described in Chapter 2.4. However, we add models for the availability of renewable energy, the pricing of renewable energy, and the social objective.

The availability of renewable energy To model the availability of renewable energy we use standard models of wind and solar from [17, 27]. Though simple, these models capture the average trends for both wind and solar accurately. Since these models are smoother than actual intermittent renewable sources, especially wind, they conservatively estimate the benefit due to following renewables.

We consider two settings (i) high wind penetration, where 90% of renewable energy comes from wind and (ii) high solar penetration, where 90% of renewable energy comes from solar. The availability given by these models is shown in Figure 3.1(a). Setting (i) is motivated by studies such as [25]. Setting (ii) is motivated by the possibility of on-site or locally contracted solar, which is increasingly common.

Building on these availability models, 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 [12].

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.

Demand response and dynamic pricing Internet-scale systems have 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 markets to reduce both their energy 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.

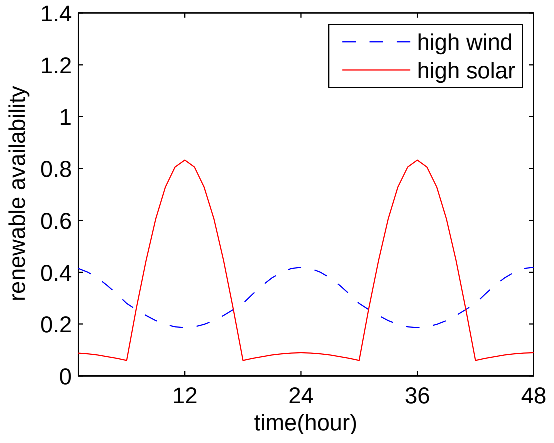
The way $p_i(t)$ is chosen as a function of $\alpha_i(t)$ will be of fundamental importance to the social impact of geographical load balancing. To highlight this, we consider a parameterized "differentiated pricing" model that uses a price p_b for brown energy and a price p_g for green energy. Specifically,

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

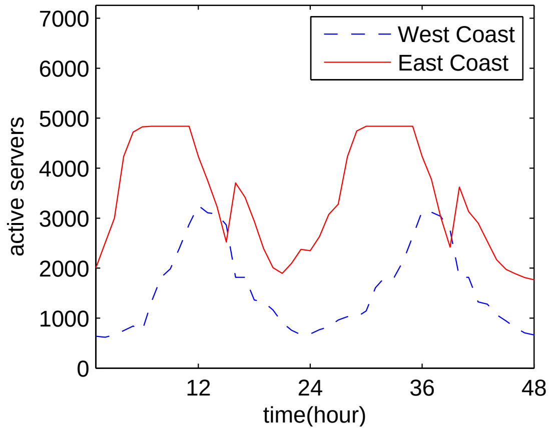
Note that $p_g = p_b$ corresponds to static pricing, and we show in the next section that $p_g = 0$ corresponds to socially optimal pricing. Our experiments vary $p_g \in [0, p_b]$.

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 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:

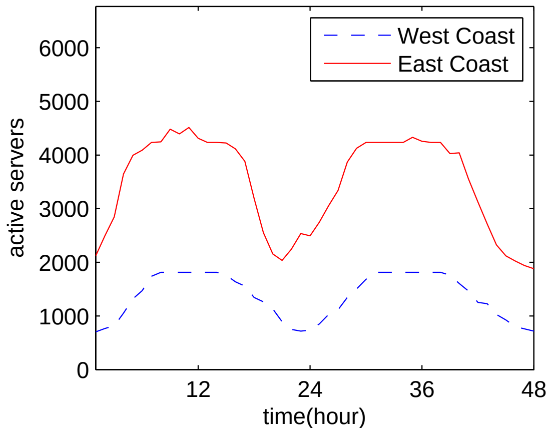
$$\min_{\mathbf{m}(t),\boldsymbol{\lambda}(t)} \sum_{t=1}^{T}\sum_{i\in N}\left((1-\alpha_i(t))\frac{\mathcal{E}_i(t)}{p_i(t)} + \tilde{\beta}\mathcal{D}_i(t)\right) \tag{3.1}$$



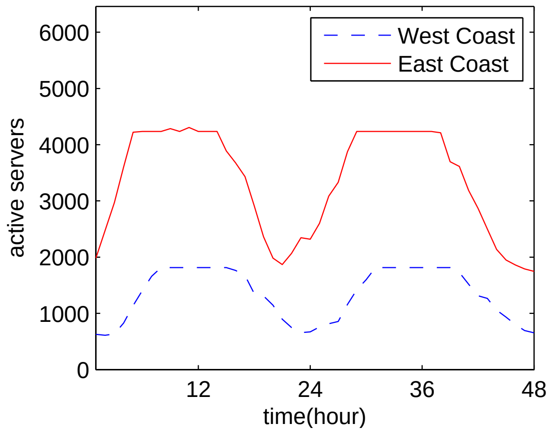
(a) Renewable availability



(b) High solar penetration



(c) High wind penetration

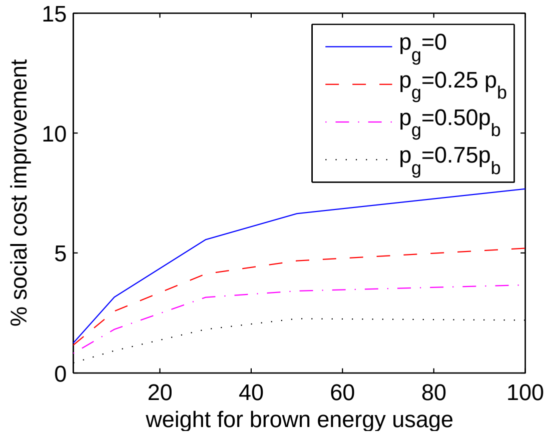


(d) Static pricing

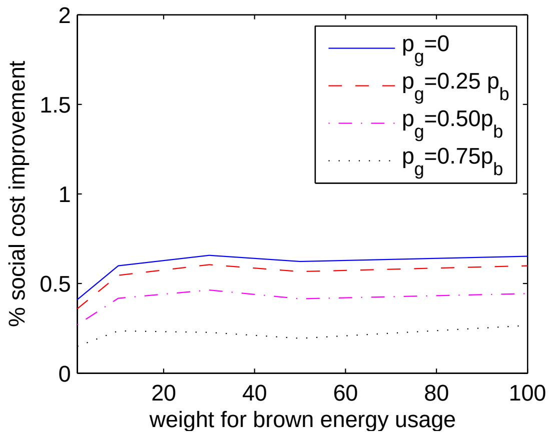
Figure 3.1: Geographical load balancing "following the renewables" under optimal pricing. (a) Availability of wind and solar. (b)–(d) Capacity provisioning of east coast and west coast data centers when there are renewables, high solar penetration, and high wind penetration, respectively.

where $\mathcal{D}_i(t)$ is the delay cost defined in (2.2), $\mathcal{E}_i(t)$ is the energy cost defined in (2.1), 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.



(a) High solar penetration



(b) High wind penetration

Figure 3.2: Reduction in social cost from dynamic pricing compared to static pricing as a function of the weight for brown energy usage, $1/\tilde{\beta}$, under (a) high solar penetration and (b) high wind penetration.

3.1.2 The importance of dynamic pricing

To begin our experiments, we illustrate that optimal pricing can lead geographical load balancing to "follow the renewables." Figure 3.1 shows this in the case of high solar penetration and high wind penetration for $\tilde{\beta} = 0.1$. By comparing Figures 3.1(b) and 3.1(c) to Figure 3.1(d), which uses static pricing, the change in capacity provisioning, and thus energy usage, is evident. For example, Figure 3.1(b) shows a clear shift of service capacity from the east coast to the west coast as solar energy becomes highly available and then back when solar energy is less available. Similarly, Figure 3.1(c) shows a shift, though much smaller, of service capacity toward the evenings, when wind is more available. 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 3.1 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, this is likely to be a slow process. Thus, it is important to consider the impact of partial adoption of dynamic pricing in addition to full, optimal dynamic pricing.

Figure 3.2 focuses on this issue. To model the partial adoption of dynamic pricing, we can consider $p_g \in [0, p_b]$. Figure 3.2(a) shows that the benefits provided by dynamic pricing are moderate but significant, even at partial adoption (high p_g), when there is high solar penetration. Figure 3.2(b) suggests that there would be much less benefit if renewable sources were dominated by wind with only diurnal variation, because the availability of solar energy is much more correlated with the traffic peaks. Specifically, the three hour gap in time zones means that solar on the west coast can still help with the high traffic period of the east coast, but the peak average wind energy is at night. However, wind is vastly more bursty than this model predicts, and a system which responds to these bursts will still benefit significantly.

Another interesting observation about the Figure 3.2 is that the curves increase faster in the range when $\tilde{\beta}$ is large, 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, but we omit the results for brevity.

3.2 Local renewables

Now we continue to investigate the case of the local renewable supply. Our numeric experiments combine analytic models with real traces for workload and renewable availability, to allow controlled experimentation but provide realistic findings. We now explain the setup, which extends that of [33].

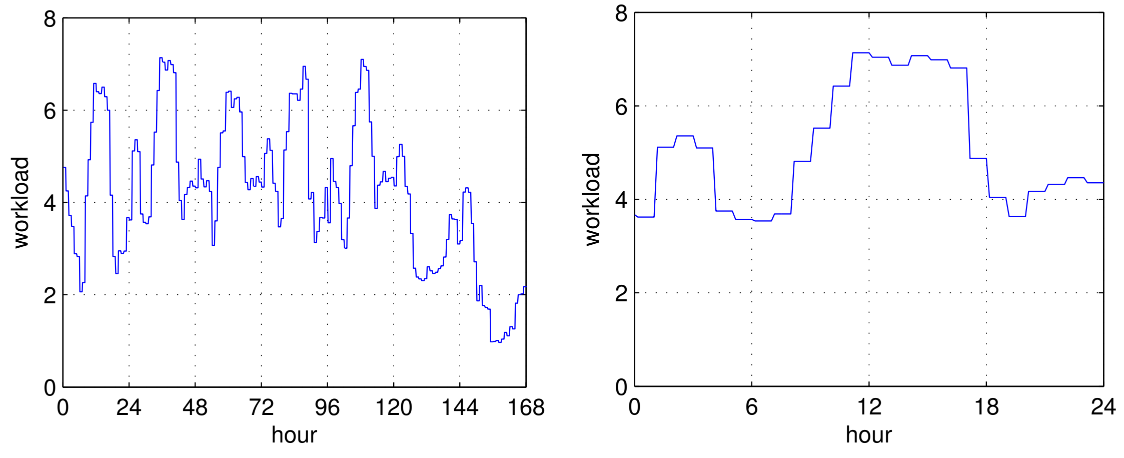
3.2.1 Experimental setup

The workload Our workload model considers a set J of sources of requests, with one source at the center of each of the 48 continental states in the US.

We consider one hour time slots for a week. At the start of each slot, the routing and capacity of each data center are updated. We use a slot length of 1 hour so that servers are not turned on and off too frequently given the significant wear-and-tear costs of power-cycling. Let $L_j(t)$ denote the mean arrival rate from source j at time t.

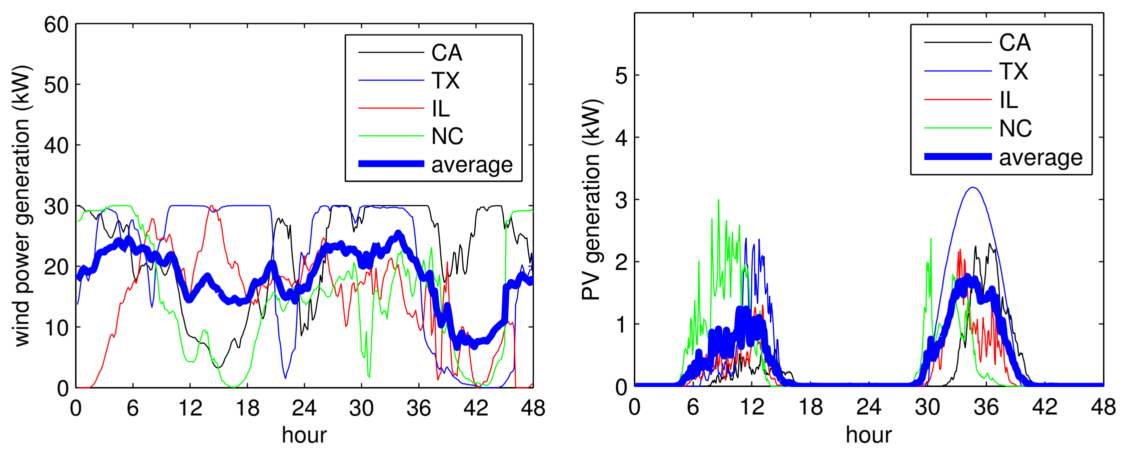
To provide realistic estimates, we use real-world traces to define $L_j(t)$. The workload of each source is scaled proportionally to the number of internet users in the corresponding state. The workload is taken from a trace at Hewlett-Packard Labs [20] and is shifted in time to account for time zone of each state. The workload per internet user used in this paper is shown in Figure 3.3.

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 [22, 23] that have measurements every 10 minutes for a year. The traces of four states (CA, TX, IL, NC) are illustrated in Figure 3.4 (GHI is normalized to have average of 1). Note that as "solar", *we only consider photovoltaic generation and not solar thermal*, because of the significant infrastructure required for solar thermal plants. Since solar thermal plants typically incorporate a day's thermal storage [38], the results would be very different if solar thermal were considered.



(a) One week workload (b) Tuesday workload

Figure 3.3: HP workload trace.



(a) Wind (b) Solar

Figure 3.4: Renewable generation for two days.

These figures illustrate two important features of renewable energy: spatial variation and temporal variation. In particular, we see that wind energy does not exhibit a clearly predictable pattern throughout the day and that there is little correlation across the locations considered. In contrast, solar energy has a predictable peak during the day and is highly correlated across the locations.

In our investigation, we scale the "capacity" of wind and solar. When doing so, we scale the availability of wind and solar linearly, which is suitable when considering scaling the size of the wind farm or solar installation. Throughout, we measure the capacity of renewables as the ratio of the average renewable generation to the minimal energy required to serve the average workload. Thus, a capacity of $c = 2$ means that the average renewable generation is twice the minimal energy required to serve the average workload. In the following we set capacity to $c = 2$ by default, but vary it in Figures 3.8–3.9.

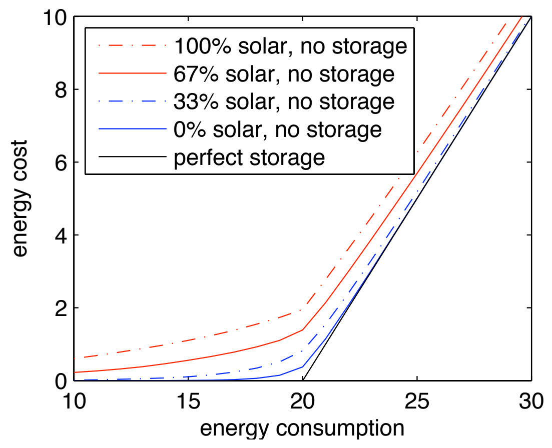


Figure 3.5: Energy cost as a function of total energy consumption, when the average renewable energy in a one-hour slot is 20.

The internet-scale system We model the internet-scale system as a set N of 10 data centers, placed at the centers of states known to have Google data centers [36], namely California, Washington, Oregon, Illinois, Georgia, Virginia, Texas, Florida, North Carolina, and South Carolina. Data center $i \in N$ contains M_i homogeneous servers, where M_i is twice the minimal number of servers required to serve the peak workload of i under a scheme which routes traffic to the nearest data center. Further, the renewable availability at each data center is defined by the trace from a nearby location; this was usually within the same state, but in five cases it was a different trace from a nearby state.

The two key control decisions of geographical load balancing are (i) determining $\lambda_{ij}(t)$, the amount of traffic routed from source j to data center i; and (ii) determining $m_i(t) \in \{0, \dots, M_i\}$, the number of active servers at data center i. The objective is to choose $\lambda_{ij}(t)$ and $m_i(t)$ to minimize the "cost" of the system. The cost is the sum of the *energy cost* and the *delay cost* (lost revenue), below. We neglect the cost of altering the number of active servers $m_i(t)$; this can be incorporated using an approach similar to that of [32].

Delay cost The delay cost captures the lost revenue incurred because of the delay experienced by the requests, where the delay includes both the network delay from source j to data center i, d_{ij}, and the queueing delay at i. We model d_{ij} to be the distance between source and data center, divided by the speed of 200 km/ms plus a constant (5 ms), resulting in delay ranging in [5 ms, 56 ms]. We model the queueing delays using parallel M/G/1/Processor Sharing queues with the total load $\lambda_i(t) = \sum_j \lambda_{ij}(t)$ divided equally among the $m_i(t)$ active servers, each having service rate $\mu_i = 0.1(\text{ms})^{-1}$. This parameter setting makes the average delay 20 ms when the utilization is 0.5, which is reasonable. Note that this model is not new, and is referred to as the GLB-LIN-Q model in [33].

Energy cost To capture the effect of integrating renewable energy, we model the energy cost as the number of active servers *excluding* those that can be powered by renewables. Note that this assumes that data centers operate their own wind and solar generations and pay no marginal cost for renewable energy. Further, it ignores the installation and maintenance costs of renewable generation.

Quantitatively, if the renewable energy available at data center i at time t is $r_i(t)$, measured in terms of number of servers that can be powered, then the energy cost is

$$p_i(m_i(t) - r_i(t))^+ \tag{3.2}$$

The p_i for each data center is constant, and equals to the industrial electricity price of each state in May 2010 [25]. This contrasts with the total power $p_i m_i$ used typically, e.g., in [33]. Although data center could instead sell the renewable supply to the grid, this will have only a small effect on p_i in a future dominated by renewable energy; when the renewable supply is high at the data center location, the local spot price of power is also likely to be low.

Storage The above formula for the energy cost is simplistic because it assumes the cost is zero if fewer servers are provisioned than the *average* renewable generation in a time slot. While this is true if one can do "perfect" smoothing of the renewables using storage, it is not necessarily true in practice. Note that the fluctuations in available power are usually considerably less than the total power draw. For our traces, "perfect" smoothing can be achieved using the amount of storage that can support the whole data center for several minutes. This is feasible, since this amount of storage is provided by currently deployed Uninterruptible Power Supplies (UPSs).

Without storage, there will be some cost incurred due to the variability of renewable availability within a time slot. Here the actual energy cost becomes

$$\mathtt{E}_\tau \left[p_i \left(m_i(t) - r_i(t,\tau) \right)^+ \right] \tag{3.3}$$

where (t, τ) is the sub-slot in time-slot t and $\mathtt{E}_\tau(r_i(t,\tau)) = r_i(t)$. Figure 3.5 shows several curves for California. The lowest curve is (3.2), i.e., the price for a system with enough storage to smooth renewable generation over one interval, and the upper ones for (3.3), i.e. the case of no storage, which is derived directly from the renewable traces as described above. The figure illustrates the curves in the case of an average renewable generation of 20 kW with $p_i = 1$, $\tau = 10$ min, and includes different mixtures of solar and wind. Interestingly, the added cost for not having storage increases as the percentage of solar increases in the renewable portfolio. The figures are almost the same for different $\mathtt{E}_\tau(r_i(t,\tau))$.

Total cost Combining the discussions above, we can now write the total cost for an internet-scale system. In particular, for the case where the data centers have small-scale storage, the cost optimization that the system seeks to solve at time t becomes:

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

$$\begin{aligned}
\text{s.t.} \quad & \sum_{i\in N} \lambda_{ij}(t) = L_j(t), && \forall j \in J && (3.4a)\\
& \lambda_{ij}(t) \geq 0, && \forall i \in N, j \in J && (3.4b)\\
& 0 \leq m_i(t) \leq M_i, && \forall i \in N && (3.4c)\\
& \lambda_i(t) \leq m_i(t)\mu_i && \forall i \in N. && (3.4d)
\end{aligned}$$

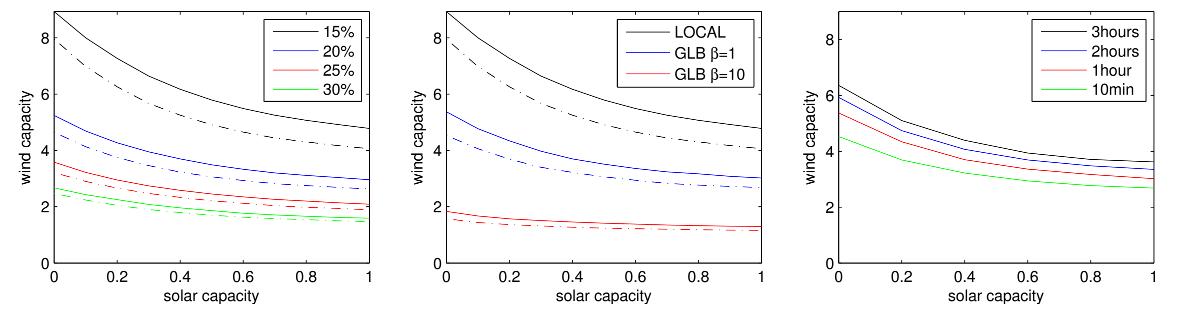
Here β determines the relative importance of energy and delay to the system. In the simulations β is set to 1 by default, but we also vary $\beta \in [1, 10]$ in Figure 3.7 to show its impact. Measurements [2] show that a 500 ms increase in delay reduces revenue by 20%, or 0.04%/ms. In a system where energy expenditure is 10% of the revenue, $\beta = 1$ or 10 corresponds to the more conservative values of 0.5 or 0.05%/ms, respectively.

When data center i can cheaply serve data from many sources, the upper bound $m_i(t) \leq M_i$ in (3.4c) can become binding, then $\lambda_{ij}(t)$ is strictly between 0 and $L_j(t)$.

When data centers have no energy storage, only the energy cost component of the optimization changes from (3.2) to (3.3). However, we cannot write the optimization in closed form for that case because the energy cost curves are determined by the renewable energy traces, as shown in Figure 3.5. To formulate and solve the optimization in this case we use a piecewise-linear approximation of the curves in Figure 3.5.

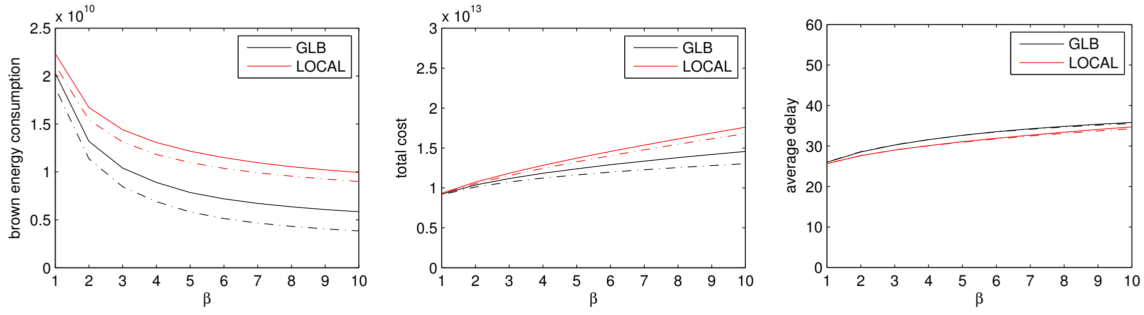
Geographical load balancing The above sections describe the cost optimization that the internet-scale system seeks to solve; however they do not describe how the system actually performs geographical load balancing to solve the optimization. In [33], decentralized algorithms are presented, which can be used to achieve the optimal cost in the setting described above. Thus, in this paper we present as "GLB" the routing and capacity provisioning decisions which solve the cost optimization problem. Note that the results do not rely on the algorithms in [33]; they simply consider the optimal allocation.

As a benchmark for comparison, we consider a system that does no geographical load balancing, but instead routes all requests to the nearest data center and optimally adjusts the number of active servers at each location. We call this system 'LOCAL' and use it to illustrate the benefits that come



(a) LOCAL for different brown energy targets (b) LOCAL and GLB for 15% brown energy target (c) GLB with different control timescales and 15% brown energy target

Figure 3.6: Wind and solar capacity required to reduce brown energy usage to 15–30% of the baseline, as a function of solar capacity. The dashed line is the same setting except with storage.



(a) Brown energy consumption (b) Total cost (c) Average delay

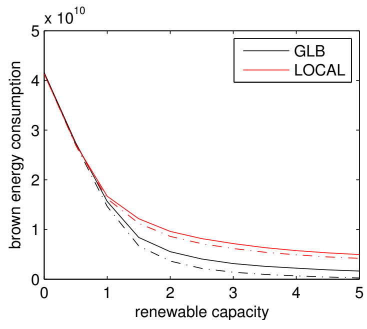
Figure 3.7: Comparison of GLB and LOCAL as a function of β. The renewable capacity is 2. The dashed line is the same setting except with storage.

from using geographical load balancing.

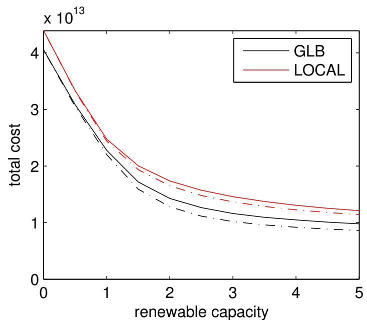
3.2.2 Results

With the setup described in the previous section, we have performed a number of numerical experiments to evaluate the feasibility of moving toward internet-scale systems powered (nearly) entirely by renewable energy. We focus on three issues: (i) the impact of geographical load balancing, (ii) the role of storage, and (iii) the optimal mix of wind and solar.

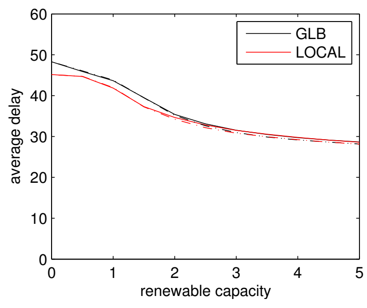
The impact of geographical load balancing Geographical load balancing is known to provide internet-scale system operators significant energy cost savings, at the expense of small increases in network delay due to the fact that requests can be routed to where energy is cheap or renewable generation is high. This behavior is illustrated in Figure 3.7, which shows the average cost and delay under GLB versus LOCAL as the cost (β) of brown energy relative to delay is increased. The novelty of Figure 3.7 is in (a), which shows the reduction in consumption of brown energy. This



(a) Brown energy consumption



(b) Total cost



(c) Average delay

Figure 3.8: Comparison of GLB and LOCAL with different renewable capacities and $\beta = 10$. The dashed line is the same setting except with storage.

illustrates that the reduction in brown consumption is significantly larger even than the reduction in cost, and that it is significant even when energy is cheap (β is small).

Next, we consider Figure 3.8, which illustrates the differences between LOCAL and GLB as a function of the capacity of renewable energy. Interestingly, Figure 3.8 highlights that when there is little capacity of renewables, both GLB and LOCAL can take advantage of it, but that as the capacity of renewables increases GLB is much more efficient at using it. This is evident because of the significantly lower brown energy consumption of GLB that emerges at capacities >1.5. Figure 3.8 also illustrates that increased capacity provides significant reductions in both the total cost and the average delay under both GLB and LOCAL.

Finally, let us consider the effect that GLB has on capacity provisioning of renewable energy. Figure 3.6 illustrates the capacity of solar and wind necessary to achieve certain brown energy reduction targets. We see in (a) that, under LOCAL, the capacity of renewables necessary to hit an 85% reduction of brown energy is extreme. However, in (b) we see that the use of GLB can provide significant reductions due to its more efficient use of the renewable capacity. If energy is expensive ($\beta = 10$), then the required capacity is nearly viable.

The discussion above highlights that GLB can be extremely useful for the adoption of renewable energy into internet-scale systems. However, there are certainly challenges that remain for the design of GLB. One particularly important challenge is that of adjusting the routing and capacity decisions at a fast enough time scale to react to the availability of renewable energy. In particular, the plots we have described so far use a control time-scale of 1 hour. If this control time-scale is slower, or if information about the availability of renewable energy is stale, then the benefits provided by GLB degrade. This is illustrated in Figure 3.6(c). This highlights the importance of reducing the energy and wear-and-tear costs of switching servers into and out of active mode, since it is the magnitude of these costs that most often limits the control time-scale.

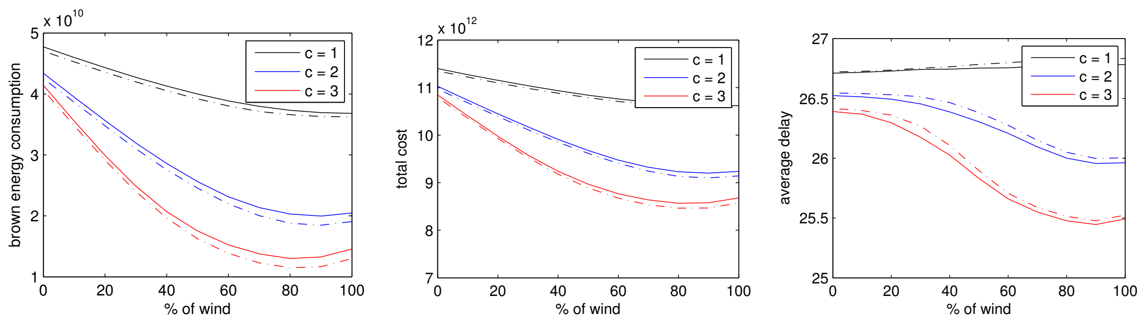
The role of storage In addition to GLB, another important tool to aid the incorporation of renewable energy into internet-scale systems is storage, such as UPSs. In this paper, we limit our consideration to small-scale storage, which is only able to power the data center over the time-scale of a few minutes. We consider small-scale storage due to the opportunity provided by the UPSs already available in data centers today, and due to the fact that the cost of large scale storage is prohibitive at this point.

Throughout Figures 3.7, 3.8, 3.6(a), 3.6(b), and 3.9, the impact of storage can be seen through the difference between the corresponding dashed and solid lines. A few trends that are evident in these plots are the following: (i) storage becomes more valuable with either higher capacities of renewables, i.e., > 1.5, or larger β; and (ii) storage plays a more significant role under GLB than under LOCAL. Both points are clearly illustrated in Figures 3.7 and 3.8, and Figure 3.8 also highlights that storage allows brown energy consumption to be almost completely eliminated when using GLB.

The optimal renewable portfolio We now move to the question of what mix of solar and wind is most effective for internet-scale systems. A priori, it seems that solar may be the most effective, since the peak of solar availability is closely aligned with that of the data center workload. However, the fact that solar is not available during the night is a significant drawback. Also, once GLB is used, it becomes possible to aggregate wind availability across geographical locations. This provides significant benefits because wind availability is not correlated across large geographical distances, and so when aggregated, the availability smoothes considerably, as illustrated in Figure 3.4(a). As a result, it seems that wind should be quite valuable to internet-scale systems.

Our results lend support to the discussion above. As illustrated in Figures 3.6 and 3.9, the optimal renewable portfolio for brown energy reduction is around 80% wind, and extra solar capacity provides little benefit beyond that point, in keeping with the findings of [21]. More specifically, Figure 3.9 shows the brown energy usage as a function of the fraction of energy coming from wind, for three values of total generating capacity, c. Keeping c fixed implicitly assumes that the cost per kWh of solar and wind installation are equal.

Interestingly, this optimal portfolio is robust to many factors including storage, renewable capacity, and even whether the system seeks to optimize brown energy consumption, total cost, or average delay. This last point is important, since it highlights that the system operators' goal is aligned with both the users' experience and society's interest. However, the optimal portfolio is affected by the workload, specifically, the peak-to-mean ratio. For large diurnal peak-to-mean ratios the optimal portfolio can be expected to use a higher percentage of solar. Another interesting direction is to include seasonal effect [21]. Here during summer the solar supply is higher and wind supply is lower, and the power demand is higher due to cooling. Therefore the optimal mix will have higher ratio of



(a) Brown energy consumption (b) Total cost (c) Average delay

Figure 3.9: Impact of the mix of renewable energy used under GLB with $\beta = 1$ when the total renewable capacity is $c = 1, 2, 3$. The dashed line is the same setting except with storage.

solar supply.

Chapter 4

Concluding remarks

This paper has focused on understanding algorithms for and the social impacts of geographical load balancing in Internet-scaled systems. We have provided two 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 energy is available. This both eases the incorporation of renewables into the grid and reduces brown energy consumption of Internet-scale systems. Last but not least, we study using local renewable to power data centers. Our numeric results highlight the importance of geographical load balancing on better utilize the renewable supply.

There are a number of interesting directions for future work that are motivated by the studies in this paper. 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 [31]? Can Internet-scale systems use energy storage at data centers in order to magnify cost reductions when participating in demand response markets? How will the cost of renewables change the optimal portfolio? Answering these questions will pave the way for greener geographic load balancing.

Bibliography

[1] US Census Bureau, http://www.census.gov.

[2] The cost of latency. *James Hamilton's Blog*, 2009.

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

[4] V. K. Adhikari, S. Jain, and Z.-L. Zhang. YouTube traffic dynamics and its interplay with a tier-1 ISP: An ISP perspective. In *ACM IMC*, pages 431–443, 2010.

[5] S. Albers. Energy-efficient algorithms. *Comm. of the ACM*, 53(5):86–96, 2010.

[6] L. L. H. Andrew, M. Lin, and A. Wierman. Optimality, fairness and robustness in speed scaling designs. In *Proc. ACM Sigmetrics*, 2010.

[7] C. L. Archer and M. Z. Jacobson. Supplying baseload power and reducing transmission requirements by interconnecting wind farms. *J. Applied Meteorology and Climatology*, 46:1701–1717, Nov. 2007.

[8] A. Beloglazov, R. Buyya, Y. C. Lee, and A. Zomaya. A taxonomy and survey of energy-efficient data centers and cloud computing systems, Technical Report, 2010.

[9] D. P. Bertsekas. *Nonlinear Programming.* Athena Scientific, 1999.

[10] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods.* Athena Scientific, 1989.

[11] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[12] S. Carley. State renewable energy electricity policies: An empirical evaluation of effectiveness. *Energy Policy*, 37(8):3071–3081, Aug 2009.

[13] 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.

[14] M. Conti and C. Nazionale. Load distribution among replicated web servers: A QoS-based approach. In *Proc. ACM Worksh. Internet Server Performance*, 1999.

[15] A. Croll and S. Power. How web speed affects online business KPIs. http://www.watchingwebsites.com, 2009.

[16] X. Fan, W.-D. Weber, and L. A. Barroso. Power provisioning for a warehouse-sized computer. In *Proc. Int. Symp. Comp. Arch.*, 2007.

[17] M. Fripp and R. H. Wiser. Effects of temporal wind patterns on the value of wind-generated electricity in california and the northwest. *IEEE Trans. Power Systems*, 23(2):477–485, May 2008.

[18] A. Gandhi, M. Harchol-Balter, R. Das, and C. Lefurgy. Optimal power allocation in server farms. In *Proc. ACM Sigmetrics*, 2009.

[19] D. Gmach, J. Rolia, C. Bash, Y. Chen, T. Christian, and A. Shah. Capacity planning and power management to exploit sustainable energy. In *Proc. of CNSM*, 2010.

[20] D. Gmach, YuanChen, A. Shah, J. Rolia, C. Bash, T. Christian, and R. Sharma. Profiling sustainability of data centers. In *Proc. ISSST*, 2010.

[21] D. Heide, L. von Bremen, M. Greiner, C. Hoffmann, M. Speckmann, and S. Bofinger. Advances in solar thermal electricity technology. *Renewable Energy*, 35(11):2483–2489, 2010.

[22] http://rredc.nrel.gov. 2010.

[23] http://wind.nrel.gov. 2010.

[24] http://www.datacenterknowledge.com, 2008.

[25] http://www.eia.doe.gov.

[26] S. Irani and K. R. Pruhs. Algorithmic problems in power management. *SIGACT News*, 36(2):63–76, 2005.

[27] T. A. Kattakayam, S. Khan, and K. Srinivasan. Diurnal and environmental characterization of solar photovoltaic panels using a PC-AT add on plug in card. *Solar Energy Materials and Solar Cells*, 44(1):25–36, Oct 1996.

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

[29] 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.

[30] M. LaMonica. Google data center to get boost from wind farm. *CNET News*, 21 April 2011.

[31] 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.

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

[33] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. H. Andrew. Greening geographic load balancing. In *Proc. ACM Sigmetrics*, 2011.

[34] 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*, pages 229–242, 2002.

[35] C. Miller. Solar-powered data centers. *Datacenter Knowledge*, 13 July 2010.

[36] R. Miller. Google data center FAQ. *Datacenter Knowledge*, 27 March 2008.

[37] R. Miller. Facebook installs solar panels at new data center. *Datacenter Knowledge*, 16 April 2011.

[38] D. Mills. Advances in solar thermal electricity technology. *solar Energy*, 76:19–31, 2004.

[39] E. Ng and H. Zhang. Predicting internet network distance with coordinates-based approaches. In *Proc. IEEE INFOCOM*, 2002.

[40] K. K. Nguyen, M. Cheriet, M. Lemay, B. St. Arnaud, V. Reijs, A. Mackarel, P. Minoves, A. Pastrama, and W. Van Heddeghem. Renewable energy provisioning for ICT services in a future internet. In *Future Internet Assembly. LNCS 6656*, pages 419–429. 2011.

[41] S. Ong, P. Denholm, and E. Doris. The impacts of commercial electric utility rate structure elements on the economics of photovoltaic systems. Technical Report NREL/TP-6A2-46782, National Renewable Energy Laboratory, 2010.

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

[43] J. Pang, A. Akella, A. Shaikh, B. Krishnamurthy, and S. Seshan. On the responsiveness of DNS-based network control. In *Proc. IMC*, 2004.

[44] M. Pathan, C. Vecchiola, and R. Buyya. Load and proximity aware request-redirection for dynamic load distribution in peering CDNs. In *Proc. OTM*, 2008.

[45] 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.

[46] 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.

[47] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[48] R. Stanojevic and R. Shorten. Distributed dynamic speed scaling. In *Proc. IEEE INFOCOM*, 2010.

[49] W. Theilmann and K. Rothermel. Dynamic distance maps of the internet. In *Proc. IEEE INFOCOM*, 2001.

[50] E. Thereska, A. Donnelly, and D. Narayanan. Sierra: a power-proportional, distributed storage system. Technical Report MSR-TR-2009-153, Microsoft Research, 2009.

[51] O. S. Unsal and I. Koren. System-level power-aware deisgn techniques in real-time systems. *Proc. IEEE*, 91(7):1055–1069, 2003.

[52] 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.

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

[54] A. Wierman, L. L. H. Andrew, and A. Tang. Power-aware speed scaling in processor sharing systems. In *Proc. IEEE INFOCOM*, 2009.

Appendices

Appendix A

Proofs for Chapter 2.2

We now prove the results from Chapter 2.2, beginning with the illuminating Karush-Kuhn-Tucker (KKT) conditions.

A.1 Optimality conditions

As GLB-Q is convex and satisfies Slater's condition, the KKT conditions are necessary and sufficient for optimality [11]; 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 (3.4c), and $\delta_{ij} \geq 0$, ν_j and σ_i be those for (3.4b), (3.4a) and (3.4d). 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:

$$\beta\left(\frac{\mu_i}{(\mu_i-\lambda_i/m_i)^2}+d_{ij}\right)-\nu_j-\delta_{ij}-\sigma_i=0 \tag{A.1}$$

$$\delta_{ij}\lambda_{ij}=0; \qquad \delta_{ij}\geq 0, \qquad \lambda_{ij}\geq 0 \tag{A.2}$$

$$\sigma_i\,(m_i\mu_i-\lambda_i)=0; \quad \sigma_i\geq 0, \quad m_i\mu_i-\lambda_i\geq 0 \tag{A.3}$$

$$\sum_{i\in N}\lambda_{ij}=L_j \tag{A.4}$$

$$p_i-\beta\left(\frac{\lambda_i/m_i}{\mu_i-\lambda_i/m_i}\right)^2+\bar{\omega}_i-\underline{\omega}_i+\sigma_i\mu_i=0 \tag{A.5}$$

$$\bar{\omega}_i(m_i-M_i)=0; \qquad \bar{\omega}_i\geq 0, \qquad m_i\leq M_i \tag{A.6}$$

$$\underline{\omega}_i m_i=0; \qquad \underline{\omega}_i\geq 0, \qquad m_i\geq 0. \tag{A.7}$$

The conditions (A.1)–(A.4) 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 (A.5), 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 (A.1), at the optimal point,

$$d_{ij}+\frac{(1+\sqrt{p_i^*/\beta})^2}{\mu_i}=d_{ij}+\frac{\mu_i}{(\mu_i-\bar{\lambda}_i)^2}=\frac{\nu_j+\delta_j}{\beta}\geq\frac{\nu_j}{\beta} \tag{A.8}$$

with equality if $\lambda_{ij}>0$ by (A.2), establishing the claim.

Note that the solution to (A.1)–(A.4) 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 $\hat{m}_i=\lambda_i(1+\sqrt{p_i/\beta})/(\mu_i\sqrt{p_i/\beta})$ of (A.5) with $\bar{\omega}_i=\underline{\omega}_i=0$.

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 (A.2), (A.4), (A.6) and (A.7) are as for GLB-Q; the stationarity conditions are:

$$\frac{\partial g_i(m_i,\lambda_i)}{\partial\lambda_i}+\beta\left(\frac{\partial\left(\lambda_i f_i(m_i,\lambda_i)\right)}{\partial\lambda_i}+d_{ij}\right)-\nu_j-\delta_{ij}=0 \tag{A.9}$$

$$\frac{\partial g_i(m_i,\lambda_i)}{\partial m_i}+\beta\lambda_i\frac{\partial f_i(m_i,\lambda_i)}{\partial m_i}+\bar{\omega}_i-\underline{\omega}_i=0. \tag{A.10}$$

Note the feasibility constraint (3.4d) of GLB-Q is no longer 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 (A.10) 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

$$\frac{\partial g_i}{\partial \lambda_i} + r(f_i + d_{ij}) + \sum_{k \in J} \lambda_{ik} r'(f_i + d_{ik}) \frac{\partial f_i}{\partial \lambda_i} - \nu_j - \delta_{ij} = 0 \tag{A.11}$$

$$\frac{\partial g_i}{\partial m_i} + \sum_{j \in J} \lambda_{ij} r'(f_i + d_{ij}) \frac{\partial f_i}{\partial m_i} + \bar{\omega}_i - \underline{\omega}_i = 0 \tag{A.12}$$

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.

A.2 Characterizing the optima

Lemma 1 will help prove the results of Chapter 2.2.

Lemma 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 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 $m_i = M_i$. Let S' be some other optimal solution.

Since the cost function is jointly convex in λ_{ij} and m_i, any convex combination of S and S' must also be optimal. Let $m_i(s)$ denote the m_i value of a given solution s. Since $m_i(S) = M_i$, we have $\lambda_i > 0$ and so the optimality of S implies f_i is finite at S and hence differentiable. By (A.10) and the continuity of the partial derivative [47, Corollary 25.51], there is a neighborhood $\mathcal{N}$ of S within which all optimal solutions have $\bar{\omega}_i > 0$, and hence $m_i(s) = M_i$ for all $s \in \mathcal{N}$. Since $S + \epsilon(S' - S) \in \mathcal{N}$ for sufficiently small ϵ, the linearity of $m_i(s)$ implies $M_i = m_i(S + \epsilon(S' - S)) = m_i(S) + \epsilon(m_i(S') - m_i(S))$. Thus $m_i(S') = m_i(S) = M_i$. □

Proof of Theorem 1. Consider first the case where there exists an optimal solution with $m_i < M_i$. By Lemma 1, $\bar{\omega}_i = 0$ 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, let the set of convex combinations of S and S' be denoted $\{s(\lambda_i)\}$, where we have made explicit the parameterization by λ_i. The convexity of each F_k in m_k and λ_k implies that $F(s(\lambda_i)) - F_i(s(\lambda_i))$ is also convex, due to the fact that the parameterization is by definition affine. Further, since F_i is strictly convex in λ_i, this implies $F(s(\lambda_i))$ is strictly convex in λ_i, and hence has a unique optimal λ_i. □

Proof of Theorem 2. The proof when $m_i = M_i$ for all optimal solutions is identical to that of Theorem 1. Otherwise, when $m_i < M_i$ in an optimal solution, the definition of $\hat{m}$ gives $\overline{\lambda}_i = \mu_i \sqrt{p_i/\beta_i}/(\sqrt{p_i/\beta_i} + 1)$ 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 K nodes has at most $K - 1$ edges.

To prove that there exists one optimal solution with acyclic graph we will inductively reroute traffic 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.

Form 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

$$\xi \left(\sum_{(j,i)\in C} r(d_{ij} + f_i(m_i, \lambda_i)) - \sum_{(i,j)\in C} r(d_{ij} + f_i(m_i, \lambda_i)) \right) \tag{A.13}$$

Next note that the multiplier $\delta_{ij} = 0$ since $\lambda_{ij} > 0$ at S. Further, the KKT condition (A.11) for stationarity in λ_{ij} can be written as $K_i + r(d_{ij} + f_i(m_i, \lambda_i)) - \nu_j = 0$, where K_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,

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

Hence, by (A.13) 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 (A.8) 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

$$c_k - c_{k'} \neq d_{k'j} - d_{kj}. \tag{A.14}$$

However, (A.8) 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 (A.14), 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 (A.3), which is false by hypothesis, and so either $\bar{\omega}_k$ or $\bar{\omega}_{k'}$ is non-zero, giving the result by (A.6). $\square$

Appendix B

Proofs for Chapter 2.3

B.1 Algorithm 1

To prove Theorem 5 we apply a variant of Proposition 3.9 of Ch 3 in [10], which gives that if

(i) $F(\mathbf{m}, \boldsymbol{\lambda})$ is continuously differentiable and convex in the convex feasible region (3.4a)–(3.4c);

(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 [10] 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.1, 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 [10], 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 (3.4d) 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 $(\lambda_i(\tau)/m_i(\tau), i \in N)$. □

B.2 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 (2.15), 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 2. *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 $F(\boldsymbol{\lambda}) \leq \phi$ implies $\boldsymbol{\lambda} \in \Lambda$. This is true because $\phi \geq F(\boldsymbol{\lambda}) \geq \Sigma_k \frac{\beta\lambda_k}{\mu_k - \lambda_k/m_k(\lambda_k)} \geq \frac{\beta\lambda_i}{\mu_i - \lambda_i/m_i(\lambda_i)} \geq \frac{\beta\lambda_i}{\mu_i - \lambda_i/M_i}, \forall i$. Therefore $\lambda_i \leq \frac{\phi}{\phi + \beta M_i} M_i\mu_i, \forall i$. Consequently, the intial point $\boldsymbol{\lambda}(0) \in \Lambda$ and the optimal point $\boldsymbol{\lambda}^* \in \Lambda$ because $F(\boldsymbol{\lambda}^*) \leq F(\boldsymbol{\lambda})$.

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 $Z^j_{ik}(\tau+1) = \lambda_{ik}(\tau) \geq 0, \forall k \neq j, \forall i$ and $\Sigma_i Z^j_{ik}(\tau+1) = \Sigma_i \lambda_{ik}(\tau) = L_k, \forall k \neq j$. From the definiition of the projection on $\hat{\Lambda}_j(\tau)$, $Z^j_{ij}(\tau+1) \geq 0, \forall i$, $\Sigma_i Z^j_{ij}(\tau+1) = L_j$, and $\Sigma_k Z^j_{ik}(\tau+1) \leq \frac{\phi}{\phi+\beta M_i} M_i\mu_i, \forall i$. 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 $\boldsymbol{\lambda}(\tau+1) = \frac{\Sigma_j \mathbf{Z}^j(\tau+1)}{|J|}$. 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 here.

Lemma 3. *For all $\boldsymbol{\lambda}^a, \boldsymbol{\lambda}^b \in \Lambda$, we have*

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

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

Proof. Following Lemma 3, here we continue to show $\left\|\nabla F(\mathbf{M}, \boldsymbol{\lambda}^b) - \nabla F(\mathbf{M}, \boldsymbol{\lambda}^a)\right\|_2 \leq K \left\|\boldsymbol{\lambda}^b - \boldsymbol{\lambda}^a\right\|_2$.

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 $\left\|\nabla^2 F(\mathbf{M}, \boldsymbol{\lambda})\right\|_2^2 \leq \left\|\nabla^2 F(\mathbf{M}, \boldsymbol{\lambda})\right\|_1 \left\|\nabla^2 F(\mathbf{M}, \boldsymbol{\lambda})\right\|_\infty = \left\|\nabla^2 F(\mathbf{M}, \boldsymbol{\lambda})\right\|_\infty^2$. The inequality is a property of norms and the equality is from the symmetry of $\nabla^2 F(\mathbf{M}, \boldsymbol{\lambda})$. Finally,

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

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

Lemma 5. *When applying Algorithm 2 to GLB-Q,*

(a) $F(\boldsymbol{\lambda}(\tau+1)) \leq F(\boldsymbol{\lambda}(\tau)) - (\frac{1}{\gamma_m} - \frac{K}{2}) \left\|\boldsymbol{\lambda}(\tau+1) - \boldsymbol{\lambda}(\tau)\right\|_2^2$, *where* $K = |J| \max_i 2(\phi + \beta M_i)^3/(\beta^2 M_i^4 \mu_i^2)$, $\gamma_m = \max_j \gamma_j$, *and* $0 < \gamma_j < \min_i \beta^2 \mu_i^2 M_i^4/(|J|(\phi + \beta M_i)^3), \forall j$.

(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 4, we know

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

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

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

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

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

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

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

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 [10, 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 5 is parallel to that of Proposition 3.3 in Ch 3 of [10], and Theorem 6 here is parallel to Proposition 3.4 in Ch 3 of [10]. 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. □