Online Algorithms for Geographical Load Balancing

Minghong Lin*, Zhenhua Liu*, Adam Wierman*, Lachlan L. H. Andrew†
*California Institute of Technology, Email: {mhlin,zhenhua,adamw}@caltech.edu
†Swinburne University of Technology, Email: landrew@swin.edu.au

***Abstract*—It has recently been proposed that Internet energy costs, both monetary and environmental, can be reduced by exploiting temporal variations and shifting processing to data centers located in regions where energy currently has low cost. Lightly loaded data centers can then turn off surplus servers. This paper studies online algorithms for determining the number of servers to leave on in each data center, and then uses these algorithms to study the environmental potential of geographical load balancing (GLB). A commonly suggested algorithm for this setting is "receding horizon control" (RHC), which computes the provisioning for the current time by optimizing over a window of predicted future loads. We show that RHC performs well in a homogeneous setting, in which all servers can serve all jobs equally well; however, we also prove that differences in propagation delays, servers, and electricity prices can cause RHC perform badly, So, we introduce variants of RHC that are guaranteed to perform as well in the face of such heterogeneity. These algorithms are then used to study the feasibility of powering a continent-wide set of data centers mostly by renewable sources, and to understand what portfolio of renewable energy is most effective.**

I. Introduction

Energy consumption of data centers is a major concern to both operators and society. Electricity for Internet-scale systems costs millions of dollars per month [1] and, though IT uses only a small percentage of electricity today, the growth of electricity in IT exceeds nearly all sectors of the economy. For these reasons, and more, IT must play its part in reducing our dependence on fossil fuels.

This can be achieved by using renewable energy to power data centers. Already, data centers are starting to be powered by a greener portfolio of energy [2], [3], [4]. However, achieving a goal of powering data centers *entirely* with renewable energy is a significant challenge due to the intermittency and unpredictability of renewable energy. Most studies of powering data centers *entirely* with renewable energy have focused on powering individual data centers, e.g., [5], [6]. 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.

The goal of this paper is twofold: (i) to illustrate that the geographical diversity of Internet-scale services can significantly improve the efficiency of the usage of renewable energy, and (ii) to develop online algorithms that can realize this potential.

Many papers have illustrated the potential for using "geographical load balancing" (GLB) to exploit the diversity of Internet-scale service and provide significant *cost savings* for data centers; see [1],[7]–[9]. The goal of the current paper is different. It is to explore the environmental impact of GLB within Internet-scale systems. In particular, using GLB to reduce *cost* can actually increase total energy usage: reducing the average price of energy shifts the economic balance away from energy saving measures. However, more positively, if data centers have local renewable energy available, GLB provides a huge opportunity by allowing for "follow the renewables" routing.

Research is only beginning to quantify the benefits of this approach, e.g., [10] and [11]. Many questions remain. For example: Does "follow the renewables" routing make it possible to attain "net-zero" Internet-scale services? What is the optimal mix of renewable energy sources (e.g. wind and solar) for an Internet-scale service? To address these questions, we perform a numerical study using real-world traces for workloads, electricity prices, renewable availability, data center locations, etc. Surprisingly, our study shows that wind energy is significantly more valuable than solar energy for "follow the renewables" routing. Commonly, solar is assumed to be more valuable given the match between the peak traffic period and the peak period for solar energy. Wind energy lacks this correlation, but also has little correlation across locations and is available during both night and day; thus the aggregate wind energy over many locations exhibits much less variation than that of solar energy [12].

Our numerical results suggest that using GLB for "follow the renewables" routing can provide significant environmental benefits. However, achieving this is a challenging algorithmic task. The benefits come from dynamically adjusting the routing and service capacity at each location, but the latter incurs a significant "switching cost" in the form of latency, energy consumption, and/or wear-and-tear. Further, predictions of the future workload, renewable availability, and electricity price are inaccurate beyond the short term. Thus online algorithms are required for GLB.

Although the distant future cannot be known, it is often possible to estimate loads a little in the future [13]. These predictions can be used by algorithms such as Receding Horizon Control (RHC), also known as Model Predictive Control, to perform geographical load balancing. RHC is commonly proposed to control data centers [14], [15] and has a long history in control theory [16]. In RHC, an estimate of the near future is used to design a tentative control trajectory; only the first step of this trajectory is implemented and, in the next time step, the process repeats.

Due to its use in systems today, we begin in Section III by analyzing the performance of RHC applied to the model of Section II. In particular, we study its competitive ratio: the worst case ratio of the cost of using RHC to the cost of using optimal provisioning based on perfect future knowledge. We prove that RHC does work well in some settings, e.g., in a homogeneous setting (where all servers are equally able to serve every request) RHC is $1+O(1/w)$-competitive, where w is the size of the prediction window. This can be much tighter than the competitive ratio of 3 obtained by schemes unaware of future loads [17]. However, in general, RHC can perform badly for the heterogeneous settings needed for geographical load balancing. In general, RHC is $1+\Omega(\beta/e_0)$-competitive, where β measures the switching cost and e_0 is the cost of

running an idle server. This can be large and, surprisingly, does not depend on w. That is, the worst case bound on RHC does not improve as the prediction window grows.

Motivated by the weakness of RHC in the general context of geographical load balancing, we design a new algorithm in Section III called Averaging Fixed Horizon Control (AFHC). AFHC works by taking the average of $w+1$ Fixed Horizon Control (FHC) algorithms. Alone, each FHC algorithm seems much worse than RHC, but by combining them AFHC achieves a competitive ratio of $1+O(1/w)$, superior to that of RHC. We evaluate these algorithms in Section IV under real data center workloads, and show that the improvement in worst-case performance comes at no cost to the average-case performance.

Note that the analysis of RHC and AFHC applies to a very general model. It allows heterogeneity among both the jobs and the servers, whereas systems studied analytically typically have homogeneous servers [17]–[19] or disjoint collections thereof [20].

II. Model

Our focus is on understanding how to dynamically provision the (active) service capacity in geographically diverse data centers serving requests from different regions so as to minimize the "cost" of the system, which may include both energy and quality of service. In this section, we introduce a simple but general model for this setting. Note that the model generalizes most recent analytic studies of both dynamic resizing within a local data center and geographical load balancing among geographically distributed data centers, e.g., including [10],[21],[17],[1].

A. The workload

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, and the mean arrival rate at time t is denoted by $\lambda_t = (\lambda_{t,j})_{j\in\{1,\dots,J\}}$, where $\lambda_{t,j}$ is the mean request rate from source j at time t. We set $\lambda_t = 0$ for $t<1$ and $t>T$. In a real system, T could be a year, a timeslot could be 10 minutes.

B. The Internet-scale system

We model an Internet-scale system as a collection of S geographically diverse data centers, where data center $s \in S$ is modeled as a collection of M_s homogeneous servers.[1] We seek the values of two key GLB parameters:

(i) $\lambda_{t,j,s}$, the amount of traffic routed from source j to data center s at time t, such that $\sum_{s=1}^{S} \lambda_{t,j,s} = \lambda_{t,j}$.

(ii) $x_t = (x_{t,s})_{s\in\{1,\dots,S\}}$, where $x_{t,s} \in \{0,\dots,M_s\}$ is the number of active servers at data center s at time t.

The objective is to choose $\lambda_{t,j,s}$ and x_t to minimize the "cost" of the system, which can be decomposed into two components:

(i) The *operating cost* incurred by using active servers. It includes both the delay cost (revenue loss) which depends on the dispatching rule through network delays and the load at each data center, and also the energy cost of the active servers at each data center with particular load.

(ii) The *switching cost* incurred by toggling servers into and out of a power-saving mode between timeslots (including the delay, migration, and wear-and-tear costs).

We describe each of these in detail below.

1) Operating cost: The operating cost is the sum of the delay cost and the energy cost. Each is described below.

Delay cost: The delay cost captures the lost revenue incurred because of the delay experienced by the requests. To model this, we define $r_t(d)$ as the lost revenue associated with a job experiencing delay d at time t, which is an increasing and convex function. The delay has 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.

We model the *network delays* by a fixed delay $\delta_{t,j,s}$ experienced by a request from source j to data center s during timeslot t. We make no requirements on the structure of the $\delta_{t,j,s}$. We assume that these delays are known within the prediction window w.

To model the *queueing delay*, we let $q_s(x_{t,s}, \sum_j \lambda_{t,j,s})$ denote the queueing delay at data center s given $x_{t,s}$ active servers and an arrival rate of $\sum_j \lambda_{t,j,s}$. Further, for stability, we must have that $\sum_j \lambda_{t,j,s} < x_{t,s}\mu_s$, where μ_s is the service rate of a server at data center s. Thus, we define $q_s(x_{t,s}, \sum_j \lambda_{t,j,s}) = \infty$ for $\sum_j \lambda_{t,j,s} \geq x_{t,s}\mu_s$.

Combining the above gives the following model for the total delay cost $\mathcal{D}_{t,s}$ at data center s during timeslot t:

$$\mathcal{D}_{t,s} = \sum_{j=1}^{J} \lambda_{t,j,s} r_t \left(q_s\left(x_{t,s}, \sum\nolimits_{j'} \lambda_{t,j',s}\right) + \delta_{t,j,s}\right). \quad (1)$$

We assume that $\mathcal{D}_{t,s}$ is jointly convex in $x_{t,s}$ and $\lambda_{t,j,s}$. Note that this assumption is 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.

Energy cost: To capture the geographic diversity and variation over time of energy costs, we let $f_{t,s}(x_{t,s}, \sum_j \lambda_{t,j,s})$ denote the energy cost for data center s during timeslot t given $x_{t,s}$ active servers and arrival rate $\sum_j \lambda_{t,j,s}$. For every fixed t, we assume that $f_{t,s}(x_{t,s}, \sum_j \lambda_{t,j,s})$ is jointly convex in $x_{t,s}$ and $\lambda_{t,j,s}$. 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 [22]. 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 [23], or as a polynomial function of the speed, see [24], [25]. One important property of $f_{t,s}$ for our results is $e_{0,s}$, the minimum cost per timeslot for an active server of type s. i.e., $f_{t,s}(x_{t,s},\cdot) \geq e_{0,s}x_{t,s}$.

The total energy cost of data center s during timeslot t is

$$\mathcal{E}_{t,s} = f_{t,s}\left(x_{t,s}, \sum\nolimits_j \lambda_{t,j,s}\right). \quad (2)$$

2) Switching cost: For the switching cost, let β_s be the cost to transition a server from the sleep state to the active state at data center s. We assume that the cost of transitioning from the active to the sleep state is 0. If this is not the case, we can simply fold the corresponding cost into the cost β_s incurred

[1]Note that a heterogeneous data center can simply be viewed as multiple data centers, each having homogeneous servers.

in the next power-up operation. Thus the switching cost for changing the number of active servers from $x_{t-1,s}$ to $x_{t,s}$ is

$$d(x_{t-1,s}, x_{t,s}) = \beta_s (x_{t,s} - x_{t-1,s})^+,$$

where $(x)^+ = \max(0, x)$. The constant β_s includes the costs of (i) the energy used toggling a server, (ii) the delay in migrating state, such as data or a virtual machine (VM), when toggling a server, (iii) increased wear-and-tear on the servers toggling, and (iv) the risk associated with server toggling. If only (i) and (ii) matter, then β_s is either on the order of the cost to run a server for a few seconds (waking from suspend-to-RAM or migrating network state [26] or storage state [27]), or several minutes (to migrate a large VM [28]). However, if (iii) is included, then β_s becomes on the order of the cost to run a server for an hour [29]. Finally, if (iv) is considered then our conversations with operators suggest that their perceived risk that servers will not turn on properly when toggled is high, so β_s may be even larger.

C. Cost optimization problem

Given the workload and cost models above, we model the Internet-scale system as a cost-minimizer. In particular, we formalize the goal of the Internet-scale system as choosing the routing policy $\lambda_{t,j,s}$ and the number of active servers $x_{t,s}$ at each time t so as to minimize the total cost during $[1, T]$. This can be written as follows:

$$\begin{aligned} \min_{x_{t,s}, \lambda_{t,j,s}} \quad & \sum_{t=1}^{T} \sum_{s=1}^{S} \mathcal{E}_{t,s} + \mathcal{D}_{t,s} + d(x_{t-1,s}, x_{t,s}) \qquad (3) \\ \text{s.t.} \quad & \sum\nolimits_{s=1}^{S} \lambda_{t,j,s} = \lambda_{t,j}, \quad \forall t, \forall j \\ & \lambda_{t,j,s} \geq 0, \quad \forall t, \forall j, \forall s \\ & 0 = x_{0,s} \leq x_{t,s} \leq M_s, \quad \forall t, \forall s \end{aligned}$$

The above optimization problem is jointly convex in $\lambda_{t,j,s}$ and $x_{t,s}$, thus in many cases the solution can be found easily *offline*, i.e., given all the information in $[1, T]$. However, our goal is to find *online* algorithms for this optimization, i.e., algorithms that determine $\lambda_{t,j,s}$ and $x_{t,s}$ using only information up to time $t + w$ where $w \geq 0$ is called the "prediction window". Based on the structure of optimization (3), we can see that $\lambda_{t,j,s}$ can be solved easily at timeslot t once $x_{t,s}$ are fixed. Thus the challenge for the online algorithms is to decide $x_{t,s}$ online.

D. Generalizations

Although the optimization problem (3) is very general already, the online algorithms and results in this paper additionally apply to the following, more general framework:

$$\begin{aligned} \min_{x_1, \ldots, x_T} \quad & \sum_{t=1}^{T} h_t(x_t) + \sum_{t=1}^{T} d(x_{t-1}, x_t) \qquad (4) \\ \text{subject to} \quad & 0 \leq x_t \in \mathbb{R}^S, \qquad x_0 = 0. \end{aligned}$$

where x_t has a vector value and $\{h_t(\cdot)\}$ are convex functions. Importantly, this formulation can easily include various SLA constraints on mean queueing delay or the queueing delay violation probability. In fact, a variety of additional bounds on x_t can be incorporated implicitly into the functions $h_t(\cdot)$ by extended-value extension, i.e., defining $h_t(\cdot)$ to be ∞ outside its domain.

To see how the optimization problem (3) fits into this general framework, we just need to define $h_t(x_t)$ for feasible x_t as the optimal value to the following optimization over $\lambda_{t,j,s}$ given $x_{t,s}$ fixed:

$$\begin{aligned} \min_{\lambda_{t,j,s}} \quad & \sum\nolimits_{s=1}^{S} (\mathcal{E}_{t,s} + \mathcal{D}_{t,s}) \qquad (5) \\ \text{s.t.} \quad & \sum\nolimits_{s=1}^{S} \lambda_{t,j,s} = \lambda_{t,j}, \quad \forall j \\ & \lambda_{t,j,s} \geq 0, \quad \forall j, \forall s \end{aligned}$$

For infeasible x_t ($x_{t,s} \notin [0, M_s]$ for some s) we define $h_t(x_t) = \infty$. We can see that the optimal workload dispatching has been captured by the definition of $h_t(x_t)$. Note that other restrictions of workload dispatching may be incorporated by the definition of $h_t(x_t)$ similarly.

Intuitively, this general model seeks to minimize the sum of a sequence of convex functions when "smooth" solutions are preferred, i.e., it is a *smoothed online convex optimization* problem. This class of problems has many important applications, including more general capacity provisioning in geographically distributed data centers, video streaming [30] in which encoding quality varies but large changes in encoding quality are visually annoying to users, automatically switched optical networks (ASONs) in which there is a cost for re-establishing a lightpath [31], and power generation with dynamic demand, since the cheapest types of generators typically have very high switching costs [32].

E. Performance metric

In order to evaluate the performance of the online algorithms we discuss, we focus on the standard notion of the *competitive ratio*. The competitive ratio of an algorithm $\mathcal{A}$ is defined as the maximum, taken over all possible inputs, of $cost(\mathcal{A})/cost(OPT)$, where $cost(\mathcal{A})$ is the objective function of (4) under algorithm $\mathcal{A}$ and OPT is the optimal offline algorithm. In the general context, the "inputs" are the functions $\{h_t(\cdot)\}$, which are able to capture the time-varying workload, electricity price, propagation delays and so on in our geographical load balancing problem.

Actually the geographical load balancing problem (3) and the generalization (4) are instances of the class of problems known as "Metrical Task Systems (MTSs)". MTSs have received considerable study in the algorithms literature, and it is known that if no further structure is placed on them, then the best deterministic algorithm for a MTS has competitive ratio proportional to the number of system states [33], which is infinity in our problem.

Note that the analytic results of Section III focus on the competitive ratio, assuming that the service has a finite duration, i.e. $T < \infty$, but allowing arbitrary sequences of convex functions $\{h_t(\cdot)\}$. Thus, the analytic results provide worst-case (robustness) guarantees. However, to provide realistic cost estimates, we also consider case studies using real-world traces for $\{h_t(\cdot)\}$ in Section IV.

III. Algorithms and results

We can now study and design online algorithms for geographical load balancing. We start by analyzing the performance of the classic Receding Horizon Control (RHC). This

uncovers some drawbacks of RHC, and so in the second part of this section we propose new algorithms which address these. We defer the proofs to Appendix.

A. Receding Horizon Control (RHC)

RHC is classical control policy [16] that has been proposed for dynamic capacity provisioning in data centers [14], [15].

Informally, RHC works by, at time τ, solving the cost optimization over the window $(\tau, \tau+w)$ given the starting state $x_{\tau-1}$. Formally, define $X^\tau(x_{\tau-1})$ as the vector in $(\mathbb{R}^S)^{w+1}$ indexed by $t \in \{\tau, \dots, \tau+w\}$, which is the solution to

$$\begin{aligned} \min_{x_\tau,\dots,x_{\tau+w}} \quad & \sum_{t=\tau}^{\tau+w} h_t(x_t) + \sum_{t=\tau}^{\tau+w} d(x_{t-1}, x_t) \\ \text{subject to} \quad & 0 \le x_t \in \mathbb{R}^S. \end{aligned} \tag{6}$$

Algorithm 1 (Receding Horizon Control: RHC)**.** *For all $t \le 0$, set the number of active servers to $x_{RHC,t} = 0$. At each timeslot $\tau \ge 1$, set the number of active servers to*

$$x_{RHC,\tau} = X^\tau_\tau(x_{RHC,\tau-1}). \tag{7}$$

In studying the performance of RHC there is a clear divide between the following two cases:

1) *The homogeneous setting ($S = 1$):* This setting considers only one class of servers, and thus corresponds to a single data center with homogeneous servers. Under this setting, only the number of active servers is important, not which servers are active, i.e., x_t is a scalar.
2) *The heterogeneous setting ($S \ge 2$):* This setting allows for different types of servers, and thus corresponds to a single data center with heterogeneous servers or to a collection of geographically diverse data centers. Under this setting, we need to decide the number of active servers of each type, i.e., x_t is a vector.

To start, let us focus on the homogeneous setting (i.e., the case of dynamic resizing capacity within a homogeneous data center). In this case, RHC performs well: it has a small competitive ratio that depends on the minimal cost of an active server and the switching cost, and decays to one quickly as the prediction window grows. Specifically:

Theorem 1. *In the homogeneous setting ($S = 1$), RHC is $(1 + \frac{\beta}{(w+1)e_0})$-competitive.*

Theorem 1 is established by showing that RHC is not worse than another algorithm which can be proved to be $(1 + \frac{\beta}{(w+1)e_0})$-competitive. Given Theorem 1, it is natural to wonder if the competitive ratio is tight. The following result highlights that there exist settings where the performance of RHC is quite close to the bound in Theorem 1.

Theorem 2. *In the homogeneous setting ($S = 1$), RHC is not better than $(\frac{1}{w+2} + \frac{\beta}{(w+2)e_0})$-competitive.*

It is interesting to note that [34] shows that a prediction window of w can improve the performance of a metrical task system by a factor of at most $2w$. If $\beta/e_0 \gg 1$ then RHC is approximately within a factor of 2 of this limit in the homogeneous case.

The two theorems above highlight that, with enough look-ahead, RHC is guaranteed to perform quite well in the homogeneous setting. Unfortunately, the story is different in the heterogeneous setting, which is required to model the geographical load balancing.

Theorem 3. *In the heterogeneous setting ($S \ge 2$), given any $w \ge 0$, RHC is $\ge (1 + \max_s(\beta_s/e_{0,s}))$-competitive.*

In particular, for any $w > 0$ the competitive ratio in the heterogeneous setting is at least as large as the competitive ratio in the homogeneous setting with no predictions ($w = 0$). Most surprisingly (and problematically), this highlights that RHC may not see any improvement in the competitive ratio as w is allowed to grow.

The proof, given in Appendix D involves constructing a workload such that servers at different data centers turn on and off in a cyclic fashion under RHC, whereas the optimal solution is to avoid such switching. Therefore, $\{h_t(\cdot)\}$ resulting in bad competitive ratio are not any weird functions but include practical cost functions for formulation (3). Note that the larger the prediction window w is, the larger the number of data centers must be in order to achieve this worst case.

The results above highlight that, though RHC has been widely used, RHC may result in unexpected bad performance in some scenarios, i.e., it does not have "robust" performance guarantees. The reason that RHC may perform poorly in the heterogeneous setting is that it may change provisioning due to (wrongly) assuming that the switching cost would get paid off within the prediction window. For the geographical load balancing case, the electricity price based on the availability of renewable power (e.g., wind or solar) may change dramatically during a short time period. It is very hard for RHC to decide which data centers to increase/decrease capacity without knowing the entire future information, thus RHC may have to change its decisions and shift the capacity among data centers very frequently, which results in a big switching cost. Notice that this does not happen in the homogeneous setting where we don't need to decide which data center to use, and the new information obtained in the following timeslots would only make RHC correct its decision monotonically (increase but not decrease the provisioning by Lemma 3).

In the rest of this section we propose an algorithm with significantly better robustness guarantees than RHC.

B. Fixed Horizon Control

In this section, we present a new algorithm, Averaging Fixed Horizon Control (AFHC), which addresses the limitation of RHC identified above. Specifically, AFHC achieves a competitive ratio for the heterogeneous setting that matches that of RHC in the homogeneous setting.

Intuitively, AFHC works by combining $w+1$ different bad algorithms, which each use a fixed horizon optimization, i.e., at time 1 algorithm 1 solves and implements the cost optimization for $[1, 1+w]$, at time 2 algorithm 2 solves and implements the cost optimization for $[2, 2+w]$, etc.

More formally, first consider a family of algorithms parameterized by $k \in [1, w+1]$ that recompute their provisioning periodically. For all $k = 1, \dots, w+1$, let $\Omega_k = \{i : i \equiv k \mod (w+1)\} \cap [-w, \infty)$; this is the set of integers congruent to k modulo $w+1$, such that the lookahead window at each $\tau \in \Omega_k$ contains at least one $t \ge 1$.

Algorithm 2 (Fixed Horizon Control, version k: FHC$^{(k)}$)**.** *For all $t \le 0$, set the number of active servers to $x^{(k)}_{FHC,t} = 0$. At*

timeslot $\tau \in \Omega_k$, for all $t \in \{\tau, \ldots, \tau + w\}$, use (6) to set

$$x^{(k)}_{FHC,t} = X^{\tau}_t \left(x^{(k)}_{FHC,\tau-1} \right). \quad (8)$$

For notational convenience, we often set $x^{(k)} \equiv x^{(k)}_{FHC}$. Note that for $k > 1$ the algorithm starts from $\tau = k - (w+1)$ rather than $\tau = k$ in order to calculate $x^{(k)}_{FHC,t}$ for $t < k$.

FHC can clearly have very poor performance. However, surprisingly, by averaging different versions of FHC we obtain an algorithm with better performance guarantees than RHC. More specifically, AFHC is defined as follows.

Algorithm 3 (Averaging FHC: AFHC)**.** *At timeslot $\tau \in \Omega_k$, use FHC$^{(k)}$ to determine the provisioning $x^{(k)}_{\tau}, \ldots, x^{(k)}_{\tau+w}$, and then set $x_{AFHC,t} = \sum_{n=1}^{w+1} x^{(n)}_t / (w+1)$.*

Intuitively, AFHC seems worse than RHC because RHC uses the latest information to make the current decision and AFHC relies on FHC which makes decisions in advance, thus ignoring some possibly valuable information. This intuition is partially true, as shown in the following theorem, which states that RHC is not worse than AFHC for any workload in the homogeneous setting ($S = 1$).

Theorem 4. *In the homogeneous setting ($S = 1$), $cost(RHC) \leq cost(AFHC)$.*

Though RHC is always better than AFHC in the homogeneous setting, the key is that AFHC can be significantly better than RHC in the heterogeneous case, even when $S = 2$.

Theorem 5. *In heterogeneous setting ($S \geq 2$), there exist convex functions $\{h_t(\cdot)\}$ such that*

$$cost(RHC) > cost(AFHC).$$

Moreover, the competitive ratio of AFHC is much better than that of RHC in the heterogeneous case.

Theorem 6. *In both the homogeneous setting and the heterogeneous setting, AFHC is $\left(1 + \max_s \frac{\beta_s}{(w+1)e_{0,s}}\right)$-competitive.*

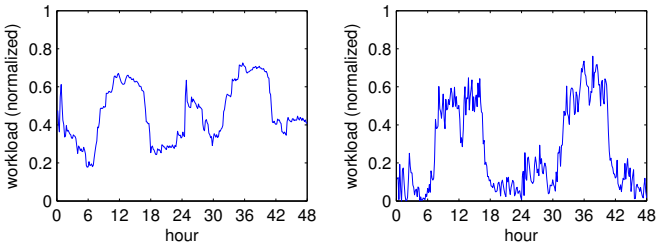
The contrast between Theorems 3 and 6 highlights the improvement AFHC provides over RHC. In fact, AFHC has the same competitive ratio in the general (possibly heterogeneous) case that RHC has in the homogeneous case. So, AFHC provides the same robustness guarantee for geographical load balancing that RHC can provide for a homogeneous local data center.

IV. Case studies

In the remainder of the paper, we provide a detailed study of the performance of the algorithms described in the previous section. Our goal is threefold: (i) to understand the performance of the algorithms (RHC and AFHC) in realistic settings; (ii) to understand the potential environmental benefits of using geographical load balancing to implement "follow the renewables" routing; and (iii) to understand the optimal portfolio of renewable sources for use within an Internet-scale system.

A. Experimental setup

This study uses the setup similar to that of [10], based on real-world traces for data center locations, traffic workloads, renewable availability, energy prices, etc, as described below.[2]



(a) Trace 1 (b) Trace 2

Fig. 1. HP workload traces.

1) The workload: We consider 48 sources of requests, with one source at the center of each of the 48 continental US states. We consider 10-minute time slots over two days.

The workload λ_t is generated from two traces at Hewlett-Packard Labs [6] shown in Figure 1. These are scaled proportional to the number of internet users in each state, and shifted in time to account for the time zone of that state.

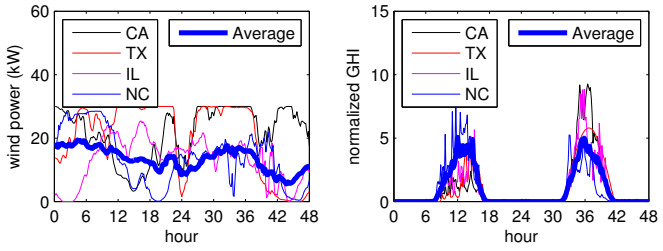
2) The availability of renewable energy: To capture the availability of solar and wind energy, we use traces with 10 minute granularity from [35], [36] for Global Horizontal Irradiance (GHI) scaled to average 1, and power output of a 30kW wind turbine. The traces of four states (CA, TX, IL, NC) are illustrated in Figure 2. Note that *we do not consider solar thermal*, because of the significant infrastructure it requires. Since these plants often incorporate a day's thermal storage [37], the results could be very different if solar thermal were considered.

These figures illustrate two important features of renewable energy: spatial variation and temporal variation. In particular, wind energy does not exhibit a clear pattern throughout the day and 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 models scaling the number of generators in a wind farm or solar installation, rather than the capacity of each. We measure the "capacity" c of renewables as the ratio of the *average* renewable generation to the minimal energy required to serve the average workload. Thus, $c = 2$ means that the average renewable generation is twice the minimal energy required to serve the average workload. We set capacity $c = 1$ by default, but vary it in Figures 5 and 7.

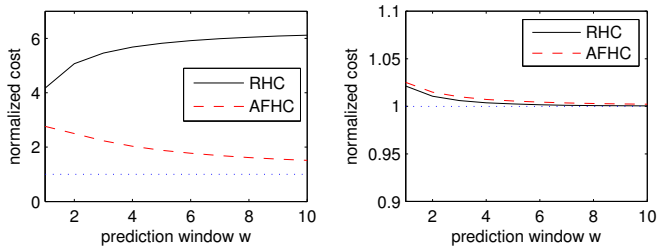
3) The Internet-scale system: We consider the Internet-scale system as a set of 10 data centers, placed at the centers of states known to have Google data centers [38], namely California, Washington, Oregon, Illinois, Georgia, Virginia, Texas, Florida, North Carolina, and South Carolina. Data center s contains M_s homogeneous servers, where M_s is set to be twice the minimal number of servers required to serve the peak workload of data center s under a scheme which routes traffic to the nearest data center. Further, the renewable availability at each data center is defined by the wind/solar trace from a nearby location, usually within the same state.

[2]Note that the setup considered here is significantly more general than that of [10], as follows. Most importantly, [10] did not model switching costs (and so did not consider online algorithms). Additionally, the current work investigates the optimal renewable portfolio more carefully, using multiple traces and varying the renewable capacity among other things.



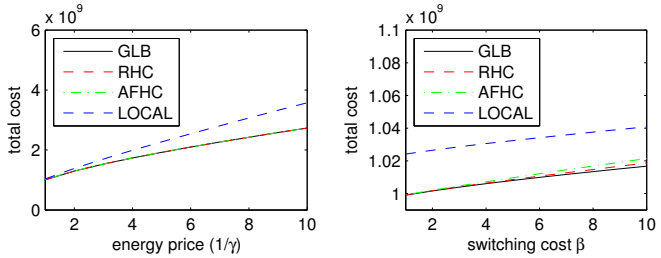
(a) Wind (b) Solar

Fig. 2. Renewable generation for two days.



(a) Cost using workload used in the proof of Theorem 3 (b) Cost using Trace 1

Fig. 3. Total cost, normalized by the cost of OPT, versus prediction window under RHC and AFHC



(a) Effect of energy price, with switching cost $\beta = 6$. (b) Effect of switching cost with delay cost $\gamma = 1$.

Fig. 4. Impact of the energy price and switching cost when the total renewable capacity is $c = 1$.

We set 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 s at time t is $r_{t,s}$, measured in terms of number of servers that can be powered, then the energy cost of data center s at time t is

$$\mathcal{E}_{t,s} = p_s(x_{t,s} - r_{t,s})^+. \tag{9}$$

Here p_s for each data center is constant, and equals to the industrial electricity price of each state in May 2010 [39]. This contrasts with the total power cost $p_s x_{t,s}$ typically used without owning renewable generation.

For delay cost, we set the round-trip network delay $\delta_{t,j,s}$ to be proportional to the distance between source and data center plus a constant (10 ms), resulting in round-trip delays in [10 ms, 260 ms]. We model the queueing delays using parallel M/GI/1/Processor Sharing queues with the total load $\sum_j \lambda_{t,j,s}$ divided equally among the $x_{t,s}$ active servers, each having service rate $\mu_s = 0.2(\text{ms})^{-1}$. Therefore, the delay cost of data center s at time t is

$$\mathcal{D}_{t,s} = \gamma \sum_j \lambda_{t,j,s} \left(\frac{1}{\mu_s - \sum_j \lambda_{t,j,s}/x_{t,s}} + \delta_{t,j,s} \right). \tag{10}$$

Here we consider linear lost revenue function $r_t(d) = \gamma d$, where γ is set to be 1. Measurements [40] show that a 500 ms increase in delay reduces revenue by 20%, or 0.04%/ms. To get a conservative estimate of the benefit from geographical load balancing, we pick $\gamma = 1$, which is slightly higher than [40], so that the penalty for the propagation delay of geographical load balancing is high compared to the benchmark policy. Later we scale p_s (with $\gamma = 1$ corresponding to the default setting) in Figure 4(a) to show the impact of relative energy cost to delay cost as energy price possibly goes high in future, or the delay penalty is lower for the systems.

For the switching cost, we set $\beta = 6$ by default, which corresponds to the operating cost of an idle server for about half an hour to one hour. We vary β in Figure 4(b) to show its impact on cost saving. For the prediction window, we set $w = 3$ by default, which corresponds to half an hour prediction of workload and renewable generation. We vary w in Figure 3 to examine its impact on cost saving.

4) Algorithms: We use optimization (3) with energy cost (9) and delay cost (10) for the geographical load balancing. We use "GLB" to denote the offline optimal solution to (3). The online solutions of algorithms Receding Horizon Control and Averaging Fixed Horizon Control are denoted by "RHC" and "AFHC", respectively.

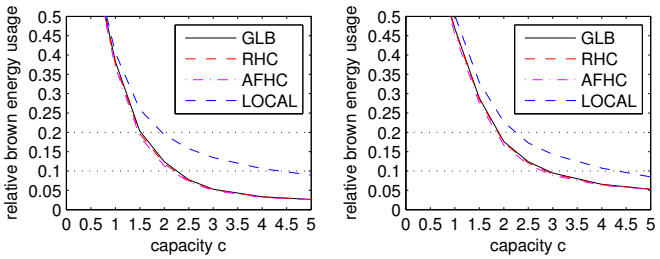
As a benchmark for comparison, we consider a system that does no geographical load balancing, but instead routes requests to the nearest data center and optimally adjusts the number of active servers at each location. We call this system 'LOCAL' and use it to illustrate the benefits that come from using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

B. Experimental results

With the foregoing setup, we performed several numerical experiments to evaluate the feasibility of moving toward Internet-scale systems powered (nearly) entirely by renewable energy and the optimal portfolios.

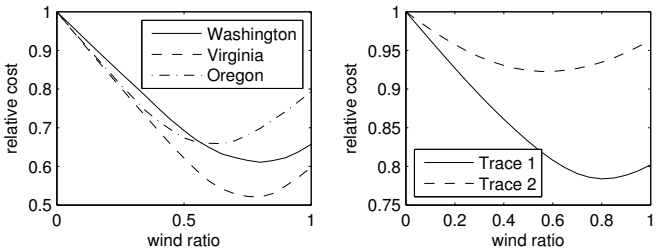
1) The performance of RHC and AFHC: Geographical load balancing is known to provide Internet-scale system operators cost savings. Let us first study the cost saving from geographical load balancing and how much of it can be achieved by the online algorithms RHC and AFHC. Figure 3(a) shows the total cost in the bad scenario with an artificial workload used in the proof of Theorem 3 (with $J = (w+1)^2$ types of jobs), which illustrates that AFHC may have much better performance in the worst case. The degradation in the performance of RHC as w grows is because J also grows. In contrast, Figure 3(b) shows the total cost of RHC and AFHC (with default settings but $\beta = 6\min(p_s)$, the same as in the bad scenario) under HP Trace 1. We can see that both RHC and AFHC are nearly optimal for the real workload. Figure 3 confirms that AFHC is able to provide worst case guarantee without giving up much performance in common cases.

This behavior under real workload is further illustrated in Figure 4, which shows the total cost under GLB, RHC, AFHC, and LOCAL as energy price or switching cost is increased. The cost saving of GLB over LOCAL becomes large when



(a) Trace 1 (b) Trace 2

Fig. 5. Impact of the renewable capacity when solar percentage is 20%.



(a) LOCAL using Trace 1 (b) GLB

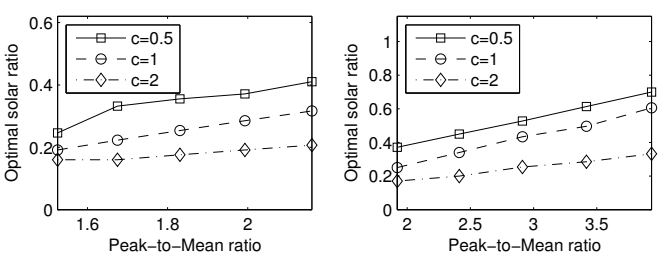
Fig. 6. Impact of the mix of renewable energy used.

the energy price is high because GLB can save a great deal of energy cost at the expense of small increases in network delay since requests can be routed to where energy is cheap or renewable generation is high. Moreover, the cost saving of GLB, RHC and AFHC over LOCAL looks stable for a wide range of switching cost.

2) The impact of geographical load balancing: Geographical load balancing is much more efficient at using renewable supply than LOCAL because it can route traffic to the data center with higher renewable generation. Figure 5 illustrates the differences of brown energy usage as a function of the capacity of renewable energy for both traces. The brown energy consumption is scaled so that the consumption is 1 when there is no renewable ($c = 0$). Interestingly, Figure 5 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, especially for Trace 1. This is evident by the significantly lower brown energy consumption of GLB that emerges at capacities > 1. For Trace 1 in Figure 5(a), the capacities of renewables necessary to reduce brown energy usage to 20% and 10% under LOCAL are 1.9 and 4.3, respectively, while those required under GLB are only 1.5 and 2.3. Similar reductions can be observed for Trace 2 in Figure 5(b).

As in Figures 3(b) and 4, the performance of RHC and AFHC is again quite close to that of the optimal solution GLB, which reinforces that both RHC and AFHC are nearly optimal in common cases. Therefore we will show only GLB and LOCAL for the remaining experiments.

3) The optimal renewable portfolio: We now move to the question of what mix of solar and wind is most effective. 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, which makes wind necessary to power the data centers during night. Our results



(a) Trace 1 (b) Trace 2

Fig. 7. Optimal portfolios for different PMRs and capacities.

lend support to the discussion above. For each data center under LOCAL, the optimal wind percentage is quite different for each location because of different renewable generation qualities, as shown in Figure 6(a). There are also similarities for different locations, e.g., the optimal portfolios contain both solar and wind, and wind has a large percentage, 60%- 90%.

Once GLB is used, it becomes possible to aggregate wind availability across geographical locations. This makes wind more valuable since wind is not correlated across large geographical distances, and so when aggregated, the availability smoothes. As illustrated in Figure 6(b), the optimal renewable portfolio for Trace 1 contains 80% wind. We can also see that the optimal portfolio is affected significantly by the workload characteristics. Compared with Trace 1, Trace 2 has less base load during night, requiring less wind generation.

The impact of workload characteristics becomes more clear in Figure 7, where we use loads $\lambda'_{j,t} = \lambda^{\alpha}_{j,t}, \alpha = 1, 1.25, 1.5, 1.75, 2$, to get different Peak-to-Mean ratios. For large diurnal Peak-to-Mean ratios the optimal portfolio can be expected to use a higher percentage of solar because solar peak is closely aligned with the workload peak, which is validated in Figure 7. Also, when renewable capacity is fairly large and we plan to install extra capacity, since solar generation can already provide enough power to serve the workload peak around noon, the increased renewable capacity can then be almost from wind generation to serve the workload during other times, especially night. This will make the solar ratio lower in the optimal portfolios, which can be seen from the lines of different renewable capacities in Figure 7.

V. Concluding Remarks

This paper studies online algorithms for geographical load balancing problem in Internet-scale systems via both theoretical analysis and trace-based experiments. We show that the classical algorithm, Receding Horizon Control (RHC), works well in homogeneous setting (where all servers are equally able to serve every request). However, in general, RHC can perform badly for the heterogeneous settings needed for geographical load balancing. Motivated by the weakness of RHC, we design a new algorithm called Averaging Fixed Horizon Control (AFHC) which guarantees good performance. We evaluate RHC and AFHC under workloads measured on a real data center. The numerical results show that RHC and AFHC are nearly optimal for our traces, which implies that the improvement in worst-case performance of AFHC comes at negligible cost to the average-case performance. The experiments also reveal vital role of geographical load balancing in reducing brown energy consumption to (nearly) zero. We also perform a detailed study on the impact of

workload characteristics and renewable capacity on the optimal renewable portfolio under GLB.

ACKNOWLEDGEMENTS

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

REFERENCES

[1] A. Qureshi, R. Weber, H. Balakrishnan, J. Guttag, and B. Maggs, "Cutting the electric bill for internet-scale systems," in *ACM SIGCOMM*, Aug. 2009, pp. 123–134.
[2] http://www.google.com/green/operations/renewable energy.html.
[3] C. Miller, "Solar-powered data centers," *Datacenter Knowledge*, 13 July 2010. [Online]. Available: http://www.datacenterknowledge.com/solar-powered-data-centers
[4] R. Miller, "Facebook installs solar panels at new data center," *Datacenter Knowledge*, 16 April 2011.
[5] 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.
[6] D. Gmach, Y. Chen, A. Shah, J. Rolia, C. Bash, T. Christian, and R. Sharma, "Profiling sustainability of data centers," in *Proc. ISSST*, 2010.
[7] E. Pakbaznia and M. Pedram, "Minimizing data center cooling and server power costs," in *Proc. of ISLPED*, 2009.
[8] L. Rao, X. Liu, L. Xie, and W. Liu, "Minimizing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment," in *IEEE INFOCOM*, 2010.
[9] R. Stanojevic and R. Shorten, "Distributed dynamic speed scaling," in *IEEE INFOCOM*, 2010.
[10] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. H. Andrew, "Greening geographical load balancing," in *Proc. ACM SIGMETRICS*, San Jose, CA, 7-11 Jun 2011, pp. 233–244.
[11] ——, "Geographic load balancing with renewables," in *Proc. ACM GreenMetrics*, San Jose, CA, 7 Jun 2011.
[12] C. L. Archer and M. Z. Jacobson, "Supplying baseload power and reducing transmission requirements by interconnecting wind farms," *J. Appl. Meteorol. Climatol.*, vol. 46, pp. 1701–1717, Nov. 2007.
[13] W. Xu, X. Zhu, S. Singhal, and Z. Wang, "Predictive control for dynamic resource allocation in enterprise data centers," in *Proc. IEEE/IFIP Netw. Op. Manag. Symp (NOMS)*, 2006, pp. 115–126.
[14] X. Wang and M. Chen, "Cluster-level feedback power control for performance optimization," in *IEEE HPCA*, 2008, pp. 101–110.
[15] D. Kusic, J. O. Kephart, J. E. Hanson, N. Kandasamy, and G. Jiang, "Power and performance management of virtualized computing environments via lookahead control," *Cluster computing*, vol. 12, no. 1, pp. 1–15, Mar. 2009.
[16] W. Kwon and A. Pearson, "A modified quadratic cost problem and feedback stabilization of a linear system," *IEEE Trans. Automatic Control*, vol. AC-22, no. 5, pp. 838–842, 1977.
[17] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," in *Proc. INFOCOM*, 2011.
[18] A. Gandhi, V. Gupta, M. Harchol-Balter, and M. Kozuch, "Optimality analysis of energy-performance trade-off for server farm management," *Performance Evaluation*, no. 11, pp. 1155–1171, Nov. 2010.
[19] J. S. Chase, D. C. Anderson, P. N. Thakar, A. M. Vahdat, and R. P. Doyle, "Managing energy and server resources in hosting centers," in *Proc. Symp. Operating System Principles*, 2001, pp. 103–116.
[20] T. Horvath and K. Skadron, "Multi-mode energy management for multi-tier server clusters," in *Proc. ACM PACT*, 2008, p. 1.
[21] B. Guenter, N. Jain, and C. Williams, "Managing cost, performance, and reliability tradeoffs for energy-aware server provisioning," in *INFOCOM, 2011 Proceedings IEEE*, Apr. 2011.
[22] S. Ong, P. Denholm, and E. Doris, "The impacts of commercial electric utility rate structure elements on the economics of photovoltaic systems," National Renewable Energy Laboratory, Tech. Rep. NREL/TP-6A2-46782, 2010.
[23] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *Proc. ACM Int. Symp. Comp. Arch.*, 2007.
[24] A. Wierman, L. L. H. Andrew, and A. Tang, "Power-aware speed scaling in processor sharing systems," in *Proc. IEEE INFOCOM*, 2009, pp. 2007–2015.
[25] L. L. H. Andrew, M. Lin, and A. Wierman, "Optimality, fairness and robustness in speed scaling designs," in *Proc. ACM SIGMETRICS*, 2010.
[26] G. Chen, W. He, J. Liu, S. Nath, L. Rigas, L. Xiao, and F. Zhao, "Energy-aware server provisioning and load dispatching for connection-intensive internet services," in *Proc. USENIX NSDI*, 2008.
[27] E. Thereska, A. Donnelly, and D. Narayanan, "Sierra: a power-proportional, distributed storage system," Microsoft Research, Tech. Rep. MSR-TR-2009-153, 2009.
[28] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proc. USENIX NSDI*, 2005, pp. 273–286.
[29] P. Bodik, M. P. Armbrust, K. Canini, A. Fox, M. Jordan, and D. A. Patterson, "A case for adaptive datacenters to conserve energy and improve reliability," University of California at Berkeley, Tech. Rep. UCB/EECS-2008-127, 2008.
[30] V. Joseph and G. de Veciana, "Variability aware network utility maximization," *CoRR*, vol. abs/1111.3728, 2011.
[31] Y. Zhang, M. Murata, H. Takagi, and Y. Ji, "Traffic-based reconfiguration for logical topologies in large-scale wdm optical networks," *Journal of lightwave technology*, vol. 23, no. 10, p. 2854, 2005.
[32] S. Kaplan, "Power plants: Characteristics and costs," *Congressional Research Service*, 2008.
[33] A. Borodin, N. Linial, and M. Saks, "An optimal on-line algorithm for metrical task system," *J. ACM*, vol. 39, no. 4, pp. 745–763, 1992.
[34] E. Koutsoupias and C. Papadimitriou, "Beyond competitive analysis," in *SIAM Journal on Computing*, 2000.
[35] http://rredc.nrel.gov/solar/new_data/confrrm/, 2010.
[36] http://wind.nrel.gov, 2010.
[37] D. Mills, "Advances in solar thermal electricity technology," *solar Energy*, vol. 76, pp. 19–31, 2004.
[38] R. Miller, "Google data center FAQ," *Datacenter Knowledge*, 27 March 2008. [Online]. Available: http://www.datacenterknowledge.com/google-data-center-faq
[39] http://www.eia.doe.gov.
[40] "The cost of latency," *James Hamilton's Blog*, 2009.

APPENDIX

A. Notation

We first introduce some additional notation used in the proofs. For brevity, for any vector y we write $y_{i..j} = (y_i, \dots, y_j)$ for any $i \leq j$.

Let x^* denote the offline optimal solution to optimization (4), and OPT be the algorithm that uses x^*. Further, let X be the result of RHC, and recall that $x^{(k)}$ is the result of FHC$^{(k)}$.

Let the cost during $[t_1, t_2]$ with boundary conditions be

$$g_{t_1,t_2}(x; x_S; x_E) = \sum_{t=t_1}^{t_2} h_t(x_t) + d(x_S, x_{t_1}) + \sum_{t=t_1+1}^{t_2} d(x_{t-1}, x_t) + d(x_{t_2}, x_E). \quad (11)$$

If x_E is omitted, then by convention $x_E = 0$ (and thus $d(x_{t_2}, x_E) = 0$). If x_S is omitted, then by convention $x_S = x_{t_1-1}$. Note that $g_{t_1,t_2}(x)$ depends on x_i only for $t_1 - 1 \leq i \leq t_2$.

For any algorithm $A \in \{RHC, FHC, AFHC, OPT\}$, the total cost is $cost(A) = g_{1,T}(x_A)$.

B. Proof of Theorems 6

Lemma 1. *Since $d(\cdot)$ satisfies triangle inequality, we have*

$$cost(FHC^{(k)}) \leq cost(OPT) + \sum_{\tau \in \Omega_k} d(x^{(k)}_{\tau-1}, x^*_{\tau-1}).$$

Proof: For every $k = 1, \dots, w+1$ and every $\tau \in \Omega_k$,

$$\begin{aligned} g_{\tau,\tau+w}(x^{(k)}) &= \sum_{t=\tau}^{\tau+w} h_t(x_t^{(k)}) + \sum_{t=\tau}^{\tau+w} d(x_{t-1}^{(k)}, x_t^{(k)}) \\ &\le \sum_{t=\tau}^{\tau+w} h_t(x_t^*) + \sum_{t=\tau+1}^{\tau+w} d(x_{t-1}^*, x_t^*) \\ &\quad + d(x_{\tau-1}^{(k)}, x_{\tau-1}^*) + d(x_{\tau-1}^*, x_\tau^*) \\ &= g_{\tau,\tau+w}(x^*) + d(x_{\tau-1}^{(k)}, x_{\tau-1}^*). \end{aligned} \tag{12}$$

Summing the above over $\tau \in \Omega_k$, establishes the lemma. ∎

Proof of Theorem 6: Substituting $d(x,y) = \beta \cdot (y-x)^+$ into Lemma 1, by the convexity of h_t (and thus $g_{1,T}$),

$$\begin{aligned} \frac{cost(AFHC)}{cost(OPT)} &\le \frac{1}{w+1} \sum_{k=1}^{w+1} \frac{g_{1,T}(x^{(k)})}{cost(OPT)} \\ \le \quad 1 + \frac{\beta \cdot \sum_{t=1}^T x_{t-1}^*}{(w+1) cost(OPT)} &\le 1 + \frac{\beta \cdot \sum_{t=1}^T x_{t-1}^*}{(w+1) \sum_{t=1}^T h_t(x_t^*)} \\ \le \quad 1 + \frac{\beta \cdot \sum_{t=1}^T x_{t-1}^*}{(w+1) e_0 \cdot \sum_{t=1}^T x_t^*} &\le 1 + \max_s \frac{\beta_s}{(w+1) e_{0,s}} \end{aligned}$$

where the second step uses Lemma 1, and the last step uses the facts that $\beta_i / e_{0,i} \le \max_s \beta_s / e_{0,s}$, and $0 \le \sum_{t=1}^T x_{t-1}^* \le \sum_{t=1}^T x_t^*$ elementwise as $x_0^* = 0$. ∎

C. Proofs of Theorems 1 and 4

The following lemma says that the optimal solution on $[i,j]$ is non-decreasing in the initial condition x_{i-1} and the final condition x_{j+1}.

Lemma 2. *Let $S = 1$. Given constants $x_{i-1}, x_{j+1} \in \mathbb{R}$, let $x^{ij} = (x_i^{ij}, \dots, x_j^{ij})$ be a vector minimizing $g_{i,j}(x; x_{i-1}; x_{j+1})$. Then for any $\hat{x}_{i-1} \ge x_{i-1}$ and $\hat{x}_{j+1} \ge x_{j+1}$, there exists a vector $\hat{x}^{ij} = (\hat{x}_i^{ij}, \dots, \hat{x}_j^{ij})$ minimizing $g_{i,j}(x; \hat{x}_{i-1}; \hat{x}_{j+1})$ such that $\hat{x}^{ij} \ge x^{ij}$.*

Proof: Since x^{ij} and $\hat{x}^{ij}$ minimize their respective objectives, we have $g_{i,j}(x^{ij}; x_{i-1}; x_{j+1}) \le g_{i,j}(\hat{x}^{ij}; x_{i-1}; x_{j+1})$ and $g_{i,j}(\hat{x}^{ij}; \hat{x}_{i-1}; \hat{x}_{j+1}) \le g_{i,j}(x^{ij}; \hat{x}_{i-1}; \hat{x}_{j+1})$. If there is an x^{ij} such that the latter holds with equality, then we can choose $\hat{x}_i^{ij} = x_i^{ij}$ and consider the problem with $g_{i+1,j}$ recursively. Otherwise, i.e., the latter is a strict inequality, summing the two inequalities and canceling terms gives

$$\begin{aligned} &(x_i^{ij} - x_{i-1})^+ + (\hat{x}_i^{ij} - \hat{x}_{i-1})^+ + (\hat{x}_{j+1} - \hat{x}_j^{ij})^+ + (x_{j+1} - x_j^{ij})^+ \\ &< (\hat{x}_i^{ij} - x_{i-1})^+ + (x_i^{ij} - \hat{x}_{i-1})^+ + (\hat{x}_{j+1} - x_j^{ij})^+ + (x_{j+1} - \hat{x}_j^{ij})^+. \end{aligned}$$

Since $\hat{x}_{i-1} \ge x_{i-1}$ and $\hat{x}_{j+1} \ge x_{j+1}$, it follows that either $x_i^{ij} < \hat{x}_i^{ij}$ or $x_j^{ij} < \hat{x}_j^{ij}$, by the submodularity of $\phi(x,y) = (x-y)^+$. In either case, we can continue recursively, considering $g_{i+1,j}$ in the former case or $g_{i,j-1}$ in the latter.

Finally we have $\hat{x}^{ij} \ge x^{ij}$. ∎

The next technical lemma says that RHC has larger solutions than related algorithms that look less far ahead.

Lemma 3. *Consider a system in the homogeneous setting ($S = 1$), and constants t, $X_{t-1} \ge \tilde{x}_{t-1} \ge 0$, and $k \in [t, t+w]$. Let $\tilde{x} = (\tilde{x}_t, \dots, \tilde{x}_k)$ minimize $g_{t,k}(x; \tilde{x}_{t-1})$, and let $X = x_{RHC}$. Then $\tilde{x} \le X_{t..k}$.*

Proof: The proof is by induction. By hypothesis, $\tilde{x}_{t-1} \le X_{t-1}$. We need to prove that if $\tilde{x}_{\tau-1} \le X_{\tau-1}$, then $\tilde{x}_\tau \le X_\tau$.

Notice that $\tilde{x}_\tau$ is the first entry of a vector minimizing $g_{\tau,k}(x; x_{\tau-1})$. Similarly X_τ is the first entry of a vector minimizing $g_{\tau;\tau+w}(x, X_{\tau-1})$. If $k = \tau+w$, we have $\tilde{x}_\tau \le X_\tau$ by Lemma 2 and the tie-break rule of RHC. Otherwise, i.e., $k < \tau + w$, we know that X_τ is the first entry of a vector minimizing $g_{\tau,k}(x; X_{\tau-1}; x'_{k+1})$ with $x'_{k+1} \ge 0$. By Lemma 2 and the RHC tie-break we again have $\tilde{x}_\tau \le X_\tau$. ∎

Next comes the first main lemma used to prove Theorem 4.

Lemma 4. *In the homogeneous setting ($S = 1$), each version k of FHC allocates fewer servers than RHC:*

$$x_{FHC}^{(k)} \le x_{RHC}. \tag{13}$$

Hence $x_{AFHC} \le x_{RHC}$.

Proof: Let $X = x_{RHC}$ be the result of RHC, and $x = x_{FHC}^{(k)}$. The proof is by induction. By definition, $x_0 = X_0 = 0$. To see that $x_{\tau-1} \le X_{\tau-1}$ implies $x_\tau \le X_\tau$, notice that x_τ is the first entry of a vector minimizing $g_{\tau,k}(x; x_{\tau-1})$ for some $k \in [\tau, \tau+w]$, with $k+1 \in \Omega_k$. The implication follows by Lemma 3 and establishes (13). The proof for x_{AFHC} is immediate. ∎

Lemma 5. *In the homogeneous setting ($S = 1$), for any given vector $x \le x_{RHC}$, we have $g_{1,T}(x_{RHC}) \le g_{1,T}(x)$.*

Proof: Denote $X = x_{RHC}$. It is sufficient to construct a sequence of vectors ξ^τ such that: $\xi^1 = x$, $\xi_t^\tau = X_t$ for $t < \tau$, and $g_{1,T}(\xi^\tau)$ is nonincreasing in τ. The sequence can be constructed inductively with the additional invariant $\xi^\tau \le X$ as follows.

At stage τ, we calculate $\xi^{\tau+1}$. Apply RHC to get $X^\tau(X_{\tau-1}) = (\tilde{x}_\tau, \dots, \tilde{x}_{\tau+w})$. Note that $\tilde{x}_\tau = X_\tau \ge \xi_\tau^\tau$ since $\xi^\tau \le X$ by the inductive hypothesis. Moreover $\tilde{x}_{\tau..\tau+w} \le X_{\tau..\tau+w}$ by Lemma 3.

If $\tilde{x}_{\tau..\tau+w} \ge \xi_{\tau..\tau+w}^\tau$ elementwise, then replace elements τ to $\tau+w$ in ξ^τ to get $\xi^{\tau+1} = (\xi_{1..\tau-1}^\tau, \tilde{x}_{\tau..\tau+w}, \xi_{\tau+w+1..T}^\tau) \ge \xi^\tau$. Then $g_{1,\tau+w}(\xi^{\tau+1}) \le g_{1,\tau+w}(\xi^\tau)$ by the optimality of $\tilde{x}_{\tau..\tau+w}$. Since $\xi_{\tau+w}^\tau \le \tilde{x}_{\tau+w}$ and $d(x,y) = \beta(y-x)^+$ is non-increasing in its first argument, we also have $g_{\tau+w+1,T}(\xi^{\tau+1}) \le g_{\tau+w+1,T}(\xi^\tau)$. Therefore, $g_{1,T}(\xi^{\tau+1}) \le g_{1,T}(\xi^\tau)$. Finally, to see that $\xi^{\tau+1} \le X$, note that $\xi^\tau \le X$ and $\tilde{x}_{\tau..\tau+w} \le X_{\tau..\tau+w}$ as remarked above.

Otherwise, let $k \in [\tau+1, \tau+w]$ be the minimum index that $\tilde{x}_k < \xi_k^\tau$. Let $\xi^{\tau+1} = (\xi_{1..\tau-1}^\tau, \tilde{x}_{\tau..k-1}, \xi_{k..T}^\tau) \ge \xi^\tau$. Note that $k \ge \tau+1$ since $\tilde{x}_\tau = X_\tau$; this ensures $\xi_t^\tau = X_t$ for $t < \tau$. Again, $\xi^{\tau+1} \le X$ as in the previous case. It remains to prove $g_{1,T}(\xi^{\tau+1}) \le g_{1,T}(\xi^\tau)$.

Let $u_A = \xi_{\tau..k-1}^\tau$, $u_B = \xi_{k..\tau+w}^\tau$, $\tilde{u}_A = \tilde{x}_{\tau..k-1}$ and $\tilde{u}_B = \tilde{x}_{k..\tau+w}$. Let vectors (u_A, u_B), $(\tilde{u}_A, u_B)$, $(u_A, \tilde{u}_B)$ and $(\tilde{u}_A, \tilde{u}_B)$ be indexed by $t \in \{\tau, \dots, \tau+w\}$. To see how replacing $\xi_{\tau\dots k-1}^\tau$ by $\tilde{x}_{\tau\dots k-1}$ affects the cost in $[1,T]$, note

$$\begin{aligned} &g_{1,T}(\xi^{\tau+1}) - g_{1,T}(\xi^\tau) \\ &= g_{\tau,\tau+w}((\tilde{u}_A, u_B)) - g_{\tau,\tau+w}((u_A, u_B)). \end{aligned}$$

Now since $\tilde{x}_k < \xi_k^\tau$, $\tilde{x}_{k-1} \ge \xi_{k-1}^\tau$ and $\phi(x,y) = (x-y)^+$ is

submodular, we have

$$\begin{aligned}&(g_{\tau,\tau+w}((\tilde{u}_A,u_B))-g_{\tau,\tau+w}((u_A,u_B)))\\&+(g_{\tau,\tau+w}((u_A,\tilde{u}_B))-g_{\tau,\tau+w}((\tilde{u}_A,\tilde{u}_B)))\\=&\beta\big((\xi_k^\tau-\tilde{x}_{k-1})^+-(\xi_k^\tau-\xi_{k-1}^\tau)^+\\&+(\tilde{x}_k-\xi_{k-1}^\tau)^+-(\tilde{x}_k-\tilde{x}_{k-1})^+\big)\\\le&0.\end{aligned}\tag{14}$$

But since $(\tilde{u}_A,\tilde{u}_B)$ optimizes (6), we have

$$g_{\tau,\tau+w}((u_A,\tilde{u}_B))-g_{\tau,\tau+w}((\tilde{u}_A,\tilde{u}_B))\ge 0.$$

Thus the first bracketed term in (14) is non-positive, whence

$$\begin{aligned}&g_{1,T}(\xi^{\tau+1})-g_{1,T}(\xi^\tau))\\\le&\,g_{\tau,\tau+w}((\tilde{u}_A,u_B))-g_{\tau,\tau+w}((u_A,u_B))\\\le&\,0.\end{aligned}$$

■

Proof of Theorem 4: By Lemma 4 and AFHC, we have $\hat{x}\le x_{RHC}$. By Lemma 5, $g_{1,T}(x_{RHC})\le g_{1,T}(\hat{x})$. ■

Proof of Theorem 1: The bound on the competitive ratio of RHC follows from Theorems 4 and 6. ■

D. "Bad" instances for Receding Horizon Control (RHC)

We now prove the lower bound results in Section III by constructing instances that force RHC to incur large costs.

Proof of Theorem 2: Consider the operating cost $h_t(x_t)=e_0x_t$ for $\lambda_t\le x_t$ and $h_t(x_t)=\infty$ for $\lambda_t>x_t$. Note that this cost function is convex. Now consider the arrival pattern $\lambda=\{\lambda_t\}_{1\le t\le T}$ where $\lambda_{k(w+2)+1}=\Lambda>0$ for $k=0,1,\dots$ and other λ_t are all 0. It is easy to see that RHC will give the provisioning $X_{k(w+2)+1}=\Lambda$ and $X_t=0$ for other t. Thus we have

$$g_{1,T}(X)=\frac{T}{w+2}\Lambda e_0+\frac{T}{w+2}\beta\Lambda.$$

Now consider another provisioning policy $\hat{x}=\{\hat{x}_t=\Lambda\}_{1\le t\le T}$. Its cost is $g_{1,T}(\hat{x})=T\Lambda e_0+\beta\Lambda$. Thus

$$\begin{aligned}&g_{1,T}(X)/g_{1,T}(x^*)\ge g_{1,T}(X)/g_{1,T}(\hat{x})\\=&\frac{e_0+\beta}{(w+2)(e_0+\beta/T)}\sim\frac{1}{w+2}+\frac{\beta}{(w+2)e_0}\end{aligned}$$

as $T\to\infty$. ■

Note that the cost function in the proof of Theorem 2 is applicable to data centers that impose a maximum load on each server (to meet QoS or SLA requirements).

Proof of Theorem 5: When $S=2$, the following geographical load balancing instance causes $cost(AFHC)<cost(RHC)$.

Choose constants $f_1>f_2$ and $\beta_1<\beta_2$ such that $(w+1)f_1<(w+1)f_2+\beta_2<(w+1)f_1+\beta_1$ and $wf_1+\beta_1<wf_2+\beta_2$. These can simultaneously be achieved by choosing an arbitrary $f_1-f_2>0$, then choosing $\beta_2-\beta_1\in(w,w+1)(f_1-f_2)$, and then $\beta_2>(w+1)(f_1-f_2)$.

Let the switching cost for data center i be β_i. Let the operating cost be $h_t(x_t)=f_1x_{t,1}+f_2x_{t,2}$ for $\lambda_t\le x_{t,1}+x_{t,2}$ and $h_t(x_t)=\infty$ for $\lambda_t>x_{t,1}+x_{t,2}$. Note that this function is convex. In this system, the servers in the second data center have lower operating cost but higher switching cost (e.g., more expensive, energy-efficient severs).

Choose constants $T>w+\max(1,\beta_2/(f_1-f_2))$, and $\Lambda>0$. Now consider the cost of schemes AFHC and RHC under the load such that: (a) $\lambda_{w+1}=0$, (b) $\lambda_t=\Lambda$ for all other $t\in[1,T]$, and (c) $\lambda_t=0$ for $t\notin[1,T]$.

Under AFHC: At time $t=1$, $FHC^{(1)}$ sees $\lambda_1,\dots,\lambda_{w+1}$ and so uses Λ servers in data center 1 for the first w timeslots and turns off all servers at timeslot $w+1$. From timeslot $t=w+2$ onwards, it sees $\lambda=\Lambda$ until time T, and so uses Λ servers in data center 2 until T.

For $2\le i\le w+1$, $FHC^{(i)}$ initially sees a window of loads in which w or fewer time slots have non-zero load, and so again chooses servers in data center 1. However, the last slot in the first window, slot $i-1$, has load Λ, and so servers remain on. In the second and subsequent windows, the cost of switching is greater than uses servers in data center 1 until T. Thus its total cost is

$$\begin{aligned}cost(AFHC)=&\frac{w}{w+1}(f_1\Lambda T+\beta_1\Lambda)\\&+\frac{1}{w+1}(f_1\Lambda w+\beta_1\Lambda+f_2\Lambda(T-w-1)+\beta_2\Lambda).\end{aligned}$$

Under RHC: RHC uses only servers in data center 1 forever, for the same reason as $FHC^{(i)}$ for $2\le i\le w+1$. Thus its total cost is

$$cost(RHC)=f_1\Lambda T+\beta_1\Lambda$$

The choice of T implies $f_1(T-w)>f_2(T-w-1)+\beta_2$, and thus $cost(AFHC)<cost(RHC)$. ■

Proof of Theorem 3: The proof will be by construction. Consider an Internet-scale system with S data centers and J types of jobs (e.g., workload from different locations). Let $J\ge S\gg w$. Let the switching cost for servers in data center s be $\beta_s=\beta_0+2\epsilon sw$. Denote the type-$j$ workload at time t by $\lambda_{t,j}$ ($j\in\{1,\dots,J\}$). Let the operating cost be $h_t(x_t)=\sum_{s=1}^S(e_0-s\epsilon+C\sum_{l=s+1}^J\lambda_{t,l})x_{t,s}$ for $\sum_{s=1}^S x_{t,s}\ge\sum_{j=1}^J\lambda_{t,j}$ and $h_t(x_t)=\infty$ otherwise, where $\epsilon>0$ is a small constant and $C>\max_s\beta_s$ is a large constant. Intuitively, this operating cost function means that servers in data center s consume a little bit more energy when s is smaller, and they are very inefficient at processing workload of types higher than s. Also, the switching cost increases slightly as s increases. This may occur if all servers use roughly the same hardware, but data center s store locally only data for jobs of types 1 to s.

Consider the workload trace which has $\lambda_{t,1}=\Lambda$ for $t=1,\dots,w+1$ and $\lambda_{t,t-w}=\Lambda$ for $t=w+2,\dots,w+S$. All the other arrival rates $\lambda_{t,j}$ are zero. Then RHC would start with Λ servers in data center 1 (the cheapest to turn on) at timeslot 1, and then at each $t\in[2,S]$ would switch off servers in data center $(t-1)$ and turn on Λ servers in data center t (the cheapest way to avoid the excessive cost of processing type t jobs using servers in data center s with $s<t$). For sufficiently small ϵ, the optimal solution always uses Λ servers in data center S for $t\in[1,w+S]$. Therefore the total costs in $[1,w+S]$ for small ϵ are $cost(RHC)=\Lambda(w+S)e_0+\Lambda S\beta_0+O(\epsilon)$ and $cost(OPT)=\Lambda(w+S)e_0+\Lambda\beta_0+O(\epsilon)$. Therefore,

$$\frac{cost(RHC)}{cost(OPT)}=1+\frac{(S-1)\beta_0}{(w+S)e_0+\beta_0}+O(\epsilon).$$

For $S\gg w$ and $Se_0\gg\beta_0$ and small ϵ, this ratio will approach $1+\beta_0/e_0$, which implies the result. ■