Minimizing Electricity Cost for Geo-Distributed Interactive Services with Tail Latency Constraint

Mohammad A. Islam
University of California, Riverside

Anshul Gandhi
Stony Brook University

Shaolei Ren
University of California, Riverside

***Abstract*—Cross-border data movement has become increasingly more costly, and even been prohibited due to data sovereignty requirements. Consequently, geo-distributed interactive services, which rely on geographically distributed data sets, is quickly emerging as an important class of workloads in data centers and resulting in soaring electricity costs. While numerous geographic load balancing (GLB) techniques exist to exploit differences in electricity prices for cost savings, they do not apply to emerging geo-distributed interactive services due to two major limitations. First, they assume that each request is processed only in one data center, whereas each geo-distributed interactive request must be processed at multiple data centers simultaneously. Second, they primarily focus on meeting average latency constraints, whereas tail latencies (i.e., high-percentile latencies) are more suitable to ensure a consistently good user experience. In this paper, we make an early effort to optimize GLB decisions for geo-distributed interactive services, exploiting spatial diversity of electricity prices to minimize the total electricity cost while meeting a tail latency constraint. Our solution employs a novel data-driven approach to determine the tail latency performance for different GLB decisions, by profiling the network latency and data center latency at a low complexity. We run trace-based discrete-event simulations to validate our design, showing that it can reduce the electricity cost by more than 7% while meeting the tail latency constraint compared to the performance-aware but cost-oblivious approach.**

I. Introduction

Operating information technology (IT) infrastructure at a global scale is becoming a norm for large IT companies. For example, Google and Facebook operate data centers all around the world, both in their own self-managed data centers and leased spaces within multi-tenant data centers [1]. Consequently, data centers have quickly emerged as major energy consumers among all industry sectors, constituting a large fraction of their operators' expenses [2]–[7].

In a geo-distributed data center system, locations can be different in terms of electricity price, available renewables, and carbon efficiency, among others. These *spatial* diversities, combined with the geographic load balancing (GLB) technique, have been exploited by many prior studies for various design purposes, such as reducing the energy cost [3], [5], [8]–[10], maximizing the utilization of renewables [11], and reducing the carbon footprint [4].

While the existing studies on GLB have made a promising progress in optimizing data center energy management, they exhibit two major limitations, and hence do not apply to many emerging geo-distributed interactive services (e.g., real-time global user/market analysis [12]).

• *Processing one request in a single data center.* Up to this point, most of the prior research on GLB [5], [8], [9], [13], [14] has considered that one incoming interactive job (e.g., a web service request) is only scheduled on *one* of the available data centers. This implicitly assumes that all the data required to process the request is available and centralized in a single data center site, and is also replicated in multiple sites. This assumption, however, fails with an increasingly higher frequency. In particular, as more and more data is generated in geo-distributed locations (e.g., smart homes, IoT applications, edge computing), making all the globally generated data available in one or more data centers for centralized processing has become very challenging, if not impossible. Concretely, the main technical challenge is the sheer volume of locally-generated data and the bandwidth required to transfer the data over a large distance (e.g., across continents). Bandwidth scarcity for wide area networks spanning several continents is only expected to become worse in the future as demand continues to grow rapidly [15]. Recent works have emphasized such technical limitations of the existing centralized approach, and have proposed techniques to instead deploy geo-distributed workload processing [15]–[17]. An additional, but increasingly stringent, obstacle that invalidates the centralized approach is the rising concern for data sovereignty/residence and privacy that has been influencing many governments to limit the transfer of data across the physical borders [18], [19]. Consequently, *geo-distributed* processing is quickly emerging as the next-generation approach to processing workloads that rely on distributed data sets [15]–[17].

• *Average latency constraint.* Interactive service providers (e.g., interactive data analytics [20], search and recommendations [21]) are increasingly using a tail latency (e.g., p95 latency, i.e., at most 5% of the requests can have a latency exceeding a certain threshold) constraint as their service level agreement (SLA), as it is more suitable than average latency to ensure consistently low latencies [22]. Nonetheless, the existing GLB literature [5], [8], [9], [13], [14] has predominantly focused on meeting the average latency (due, in part, to the analytical convenience of modeling the service as a simple queueing process).

In this paper, we make an early effort to overcome the limitations in the current GLB literature to accommodate the emerging geo-distributed interactive services. Specifically, we consider an interactive service (e.g., worldwide market analysis, interactive global data analytics [16], [17], [23], among others) which relies on geo-distributed data sets.

We illustrate in Fig. 1 an overview of typical geo-distributed interactive services. The global data is spread across multiple regions (each consisting of several data centers), and data

978-1-5090-5117-5/16/$31.00

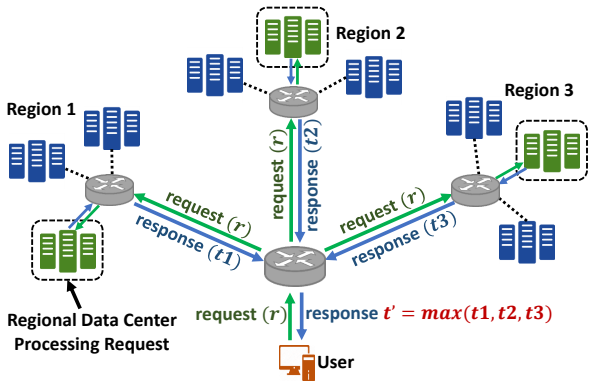


Fig. 1. Geo-distributed interactive service. Each request r is sent to multiple regions. One data center, among possibly multiple available, processes the workload at each region (green colored). The response time is determined by the slowest response.

within each region is replicated across multiple data centers in that region for fault tolerance and high availability. Thus, a user request/job needs to be sent to all regions *simultaneously*; however, only one data center in each region needs to be selected for processing the request. In practice, a region can correspond to a large country/continent (e.g., U.S., China, Europe), within which moving data across different data center locations is easier (due to a relatively shorter distance than inter-region data movement) *and* meets the data residency requirement.

While geo-distributed interactive services are quickly growing, minimizing their electricity cost subject to a tail latency constraint is challenging. Specifically, as illustrated in Fig. 1, user requests originate from different traffic sources, and each request needs to be processed in multiple data centers. Thus, unlike in the prior GLB research [5], [8], [9], [13], [14], in our problem, one request will affect the energy cost of multiple data centers. More importantly, the response time for each request is determined by the slowest response among all the data centers to which this request is sent. Consequently, request processing involves a very complex interdependency across multiple data centers, and the existing modeling techniques, such as a simple M/M/1 queueing process [9], [10], that capture the average latency, are not applicable in our problem setting. Thus, meeting the tail latency is significantly more challenging than the widely-considered average latency constraint in the existing GLB research. In fact, minimizing/meeting the tail latency within *one* data center without involving the large network latencies is already known to be a challenging problem [22], let alone in a geo-distributed setting that our study focuses on.

To address these challenges, we propose a novel geo-distributed interactive service management framework, called McTail (**M**inimizing electricity **c**osts with a **Tail** latency constraint). McTail includes a GLB optimizer that optimally splits the user traffic from each source to different geo-distributed data centers to minimize the total electricity cost. To determine the tail latency, McTail employs a data-driven approach that profiles the probability distribution of requests' response times at runtime. To avoid the curse of dimensionality due to the fact that each request needs to be processed in multiple data centers, McTail uses a scalable approach to determine the tail latency: it first profiles the latency statistics for each network path between each traffic source and each data center, as well as the latency statistics within each data center; then it exploits the latency *independence* property (as detailed in Section III-C) and calculates the end-to-end tail latency in a computationally efficient manner. Finally, we conduct trace-based simulations on an event-based simulator to validate McTail, demonstrating that McTail can reduce the energy cost by more than 7% while meeting the tail latency constraint, compared to the state-of-the-art performance-aware, but cost-oblivious, solution.

To summarize, this paper represents an early effort to optimize GLB decisions that minimize the electricity cost for geo-distributed interactive services subject to a tail latency constraint. Concretely, we consider a novel setting of geo-distributed interactive services that are in line with the emerging inter-continental bandwidth constraints and data residency requirements. We also propose an efficient algorithm, McTail, that can determine the tail latency and meet the latency constraint, while exploiting price diversity in different data centers for cost savings. Finally, we employ an event-based simulation to validate McTail.

II. Basic Problem Formulation

We consider an interactive service provider that operates N data centers around the world, represented by set $\mathcal{N}$. There are M different geographical regions, and we represent the subset of data centers located in region m by $\mathcal{N}_m$, where $m = \{1, 2, \cdots, M\}$. Note that $\sum_{i=1}^{m} |\mathcal{N}_m| = N$, where $|\mathcal{N}_m|$ denotes the number of data centers in region m. Like in the prior GLB literature [9], [10], [13], we consider a time-slotted model where each time slot corresponds to a decision epoch (e.g., 15 minutes). As the processing time for each interactive request typically takes no more than a few seconds (much less than a time slot), we omit the time indices for our time-slotted model and update the GLB decision every time slot. Other services (e.g., batch workloads) are assumed to be processed by separate systems orthogonal to our study.

Energy model. As widely used and validated by the existing GLB literature [3], [9], [24], the total energy consumption at data center j (including all IT equipment such as servers and switches) is expressed using a linear function of its total workload as

$$e_j(a_j) = e_j^{static} + e_j^{dynamic} \cdot \frac{a_j}{\mu_j}, \tag{1}$$

where e_j^{static} and $e_j^{dynamic}$ are the static and dynamic energy consumption, respectively, a_j is the total workload of data center j (measured in the same unit as the processing capacity, e.g., request/second), and μ_j is the processing capacity of data center j. Note that the network/routing devices located along the network routes (and outside the considered data centers) are typically operated by third-party Internet service providers and hence their energy consumption is not included in our model.

Tail latency model. We consider a tail end-to-end (between the traffic source and the data center) latency performance constraint at each front-end gateway or concentrated traffic source as illustrated in Fig. 1. Specifically, the high-percentile end-to-end latency (e.g., 95-percentile or p95) of requests originating from each source must be no greater than a threshold D_i. In other words, if $x\%$ is the percentile requirement, then at least $x\%$ of the requests must have an end-to-end latency not exceeding D_i.

Considering S different traffic sources, the tail latency performance of source i can be expressed as $p_i(\vec{a}, \vec{r})$, a function of data center workload $\vec{a} = \{a_1, a_2, \cdots, a_N\}$, and network route/path $\vec{r} = \{r_{i,1}, r_{i,2}, \cdots, r_{i,N}\}$, where $r_{i,j}$ denotes the network route from source i to data center j. Note that p_i represents the probability $\Pr(d_i \leq D_i)$ that the end-to-end response time d_i for requests from traffic source i does not exceed the target response time D_i.

Problem formulation. Mathematically, the operator has the following GLB optimization problem:

$$\textbf{GLB-1:} \quad \underset{\vec{a}}{\text{minimize}} \quad \sum_{j=1}^{N} [q_j \cdot e_j(a_j)] \tag{2}$$

$$\text{s.t.} \quad p_i(\vec{a}, \vec{r}) \geq P_i^{SLA}, \forall i = 1, \cdots, S \tag{3}$$

$$a_j \leq \mu_j, \forall j \in \mathcal{N} \tag{4}$$

where q_j is the electricity price at data center j and P_i^{SLA} is the SLA target (e.g., 95% for p95 latency constraint) for source i. Note that q_j may be varying over different time slots, as decided by local utilities. The objective function (2) is the total electricity cost across all data centers, (3) expresses the performance constraint set by the SLA, and (4) ensures that the total workload sent to a data center does not exceed its capacity. If network bandwidth limitations are considered, additional constraints can be included to limit the amount of traffic sent to each data center. Note that the performance constraint in (3) is equivalent to the tail latency constraint "p95 latency $\leq D_i$". In this paper, we use $p_i(\vec{a}, \vec{r}) = \Pr(d_i \leq D_i) \geq P_i^{SLA}$ (where d_i is the end-to-end latency for requests from source i), which will allow us to conveniently present the design of McTail.

III. The Design of McTail

We now present the design of McTail, which minimizes energy costs for geo-distributed services with a tail latency constraint. First, we will refine the GLB formulation to account for: (i) geo-distributed processing, and (ii) tail latency constraint (Section III-A). We then outline our solution (Section III-B) and discuss the latency profiling technique (Section III-C).

A. Problem reformulation

While we lay down the basic problem formulation in Section II, we have yet to specify our GLB decisions for geo-distributed processing and tail latency modeling, both of which are crucial for our problem setting.

GLB with geo-distributed processing. A key novelty in our study is that each user request is simultaneously sent to one data center in *each* of the M geographical regions for processing. That is, each request is sent to a *group* of M data centers (called data center group).

Recall that each region m has a set $\mathcal{N}_m$ of data centers. Thus, we have $G = \prod_{m=1}^{M} |\mathcal{N}_m|$ possible data center groups for a request, where each group consists of one data center from each region. Note that this assumes regional data replication across data centers in that region (subject to data residency requirement). If data is not replicated, the model can be adapted by considering each un-replicated data center as a separate region with only one data center.

At the traffic source i, we have a load distribution decision vector $\vec{\lambda}_i = \{\lambda_{i,1}, \lambda_{i,2}, \cdots, \lambda_{i,G}\}^T$, where $\lambda_{i,g} \geq 0$ denotes the amount of requests sent to group g from source i, and $\Lambda_i = \sum_{g=1}^{G} \lambda_{i,g}$ is the total workload from source i. Hence, the total workload sent to data center j can be expressed as:

$$a_j = \sum_{i=1}^{S} \sum_{g \in \mathcal{G}_j} \lambda_{i,g}, \tag{5}$$

where $\mathcal{G}_j$ represents the set of data center groups that have data center j as an element.

The GLB decision now is to determine the load distribution, $\vec{\lambda}_i$, for all sources. Considering all the traffic sources, we can define the load distribution matrix $\boldsymbol{\lambda} = \{\vec{\lambda}_1, \vec{\lambda}_2, \cdots, \vec{\lambda}_S\}$, which is the main decision variable in our problem. Thus, our problem of deciding the workload distribution to each *group* of data centers generalizes the existing GLB literature that only decides workload distribution to each *single* data center [3], [9], [24].

Tail latency constraint. A key challenge in our problem is how to determine the tail latency for each traffic source. To meet the tail latency constraint in (3), we need to examine each path between a source and a data center. Since we have S sources and N data centers, there are a total of $R = S \times N$ routes, each representing a network path from a source location to a data center location. We represent the route from source i to data center j by $r_{i,j}$. The end-to-end response time of a request along a certain route includes the network latency and the latency incurred *within* the data center (which we call data center latency).

In this paper, we focus on data center-level GLB decisions, while treating the scheduling decisions *within* each data center as orthogonal decisions (interested readers can refer to [22] and references therein for more details of scheduling techniques within data centers). As such, the decision under consideration that affects a data center latency is GLB, (or, equivalently, the total amount of workload sent to a data center). Hence, we represent by $p_{i,j}^{route}(a_j, r_{i,j})$ the probability that response time is less than D_i for route $r_{i,j}$, given workload a_j at data center j.

It is non-trivial to meet the tail latency constraint since each request needs to be processed in a group of data centers. Nonetheless, we make an observation that the end-to-end response time of requests sent along one route is practically *independent* of that along another route. The reason is that each interactive request is small (taking no more than a few

seconds to complete), and data centers in different regions ha different data sets. These facts, combined with other rand factors (e.g., performance interference from other workloa system slowdown and warm-up at random times [21]), lead the consequence that latencies incurred in different data cent can be viewed as uncorrelated and independent. In fact, ev for the same interactive service request processed at differ servers (but still with homogeneous settings) within the sa data center, the processing times can vary widely, by as mu as 10× [21], [25]. Additionally, considering that the netw latencies for different "source-data center" routes depend many other factors (e.g., traffic from other irrelevant service the response times for a request along different routes for g distributed processing can be viewed as independent. Note, however, that this observation may not hold for large jobs (e.g., Hadoop-based batch data processing), whose completion times primarily depend on the input data size [26] and hence have *correlated* response times in different data centers. This is left as our future work, while we focus on small interactive services in this paper.

Based on the above latency *independence* property, we can combine the response time probabilities along different routes to express $p_{i,g}^{group}(\vec{a},\vec{r})$ for requests from each source i to *each* data center group g as:

$$p_{i,g}^{group}(\vec{a},\vec{r}) = \prod_{j\in\mathcal{J}} p_{i,j}^{route}(a_j, r_{i,j}), \tag{6}$$

where $\mathcal{J}$ is the set of data centers that are in the data center group g. For example, if the data center group consists of two data centers meeting the latency threshold with probabilities of 0.99 and 0.98, respectively, then the probability that a request sent to *both* data centers will meet the latency threshold is simply $0.99 \times 0.98 \approx 0.97$.

Further, since requests from source i are distributed among multiple data center groups, the overall probability $\Pr(d_i \leq D_i)$ for requests from source i should be averaged across all the involved data center groups and hence be expressed as

$$p_i(\boldsymbol{\lambda}) = p_i(\vec{a},\vec{r}) = \frac{1}{\Lambda_i}\sum_{g=1}^{G} \lambda_{i,g} \cdot p_{i,g}^{group}(\vec{a},\vec{r}) \tag{7}$$

where we use $p_i(\boldsymbol{\lambda}) = p_i(\vec{a},\vec{r})$ to emphasize that the latency threshold satisfaction probability is a function of our GLB decision.

Reformulated problem. We now reformulate the problem **GLB-1** to explicitly account for geo-distributed processing and tail latency constraints as

$$\textbf{GLB-2:} \quad \underset{\boldsymbol{\lambda}}{\text{minimize}} \quad \sum_{j=1}^{N} [q_j \cdot e_j(a_j)] \tag{8}$$

$$\text{s.t.} \quad p_i(\boldsymbol{\lambda}) \geq P_i^{SLA}, \forall i = 1,\cdots,S, \tag{9}$$

$$\sum_{g=1}^{G} \lambda_{i,g} = \Lambda_i, \forall i = 1,\cdots,S, \tag{10}$$

$$a_j \leq \mu_j, \forall j \in \mathcal{N}, \tag{11}$$

where the constraint (10) ensures that all requests from a traffic source are processed.

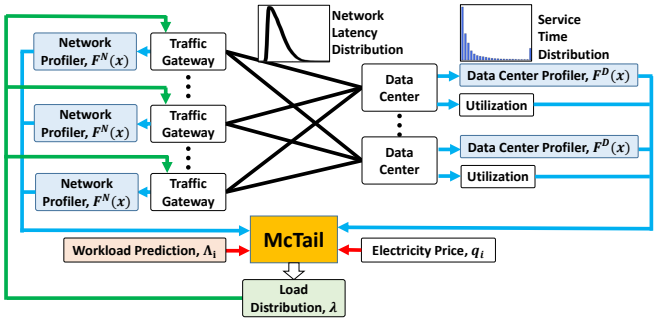


Fig. 2. Overview of McTail. Network latency and data center latency are profiled separately and sent to McTail periodically.

B. *Overview of* McTail

We show the overview of McTail in Fig. 2. The input to McTail includes the profiled network latency and data center latency distributions, the estimated workload arrival at each source during the current time slot, and the electricity price in each data center location. Then, McTail solves the problem **GLB-2** using a numerical optimization method [27] and outputs the *optimized* GLB decisions that split the incoming workloads at each source to different data center groups for geo-distributed processing. Note that if the inputs to McTail, e.g., profiled latencies, are inaccurate, the resulting GLB decisions may not be optimal. Nonetheless, our evaluation under realistic settings show that McTail is fairly robust against inaccurate inputs (see Fig. 9 for details).

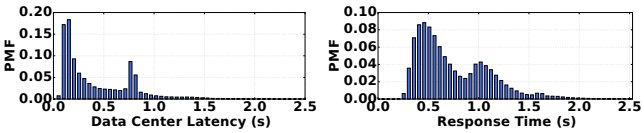
A key component of McTail is the latency profiler that determines the tail latency performance, as discussed below.

C. *Latency performance profiling*

Up to this point, we have decomposed $p_i(\boldsymbol{\lambda})$, the probability that the latency is less than the threshold D_i for each source, into (7). In order to solve **GLB-2**, we still need to determine $p_{i,j}^{route}(a_j, r_{i,j})$, i.e., the probability that response time for requests along the route from source i to data center j is less than threshold D_i.

Unfortunately, unlike the average latency that can be obtained based on simple queueing-theoretic analysis [9], [10], there is no simple closed-form expression for *tail* latency, especially when the service time of each request is not exponentially distributed. Thus, we resort to a data-driven approach by profiling the response time statistics for each route.

One approach is to directly profile the end-to-end response time statistics given W levels of workload for each route, where W represents the decision resolution and a larger W means the latency model is finer grained (in our evaluation, $W = 10$ is already a good setting). Using this approach, we need $S \times N \times W$ profiled distributions in total, which may require considerable historical data for accurate profiling. Further, the response time distributions need to be updated whenever the latency, either network latency or data center latency, changes significantly.



(a) (b)

Fig. 3. Data center latency and end-to-end Chile-Shanghai response time distributions under 30% data center utilization.

In this paper, we reduce the amount of profiled data by further decomposing the end-to-end response time for each route into two parts: network latency and data center latency (i.e., latency within a data center, including queueing time and service time). The network latency is route-dependent but workload-independent, whereas data center latency is route-independent but workload-dependent. The reason for workload-independent network latency is that it is primarily affected by the long-distance backbone network capacity, and the traffic of our considered interactive service is negligible compared to the total traffic sharing the same link.

Consequently, we denote the profiled network latency for route $r_{i,j}$ as $F^N_{i,j}$, and the data center latency distribution as $F^D_j(x)$, for data center j given its total workload of x. Then, the end-to-end latency distribution for $r_{i,j}$ becomes

$$F^R_{i,j}(x) = F^N_{i,j} * F^D_j(x), \tag{12}$$

where "$*$" denotes the convolution operator. Then, from (12), we can easily calculate $p^{route}_{i,j}(a_j, r_{i,j})$, which is the basis for determining the tail latency performance $p_i(\boldsymbol{\lambda})$ via (6) and (7).

To obtain the tail latency performance $p_i(\boldsymbol{\lambda})$ using our decomposition approach, we only need to profile $S \times N$ network latency distributions and $N \times W$ data center latency distributions. This results in significantly less profiling complexity than directly profiling each end-to-end route latency distribution (which needs $S \times N \times W$ profiled distributions).

Profiling overhead. Profiling the network and data center latency distributions in McTail does not represent a significant overhead in the existing system. In fact, existing geo-distributed data center management systems are often performance-driven and hence already closely monitor the runtime performance of different components at an even finer granularity (e.g., inside the server and network) [**?**], collecting enough information for our purposes.

Even compared to the existing GLB literature [3], [4], [9], [10] that focuses on cost minimization subject to average latency constraints, we need a comparable amount of information for implementing McTail. The difference is that the existing GLB approaches often approximate the data center *average* latency using simple queueing-theoretic modeling and profiling of the service rate in each data center [3], [4], [9], [10], whereas we adopt a data-driven approach to capture *tail* latency, which is more appropriate to ensure a consistently satisfactory performance for real-world applications.

In practice, the data center latency distribution does not vary frequently as long as the workload composition does not change significantly, while the network latency distribution may vary more frequently (due to uncontrollable external factors [28]) but is already being closely monitored by service providers. Thus, similar to existing GLB research [3], [4], [9], [10], McTail can be applied by periodically updating the latency distributions without introducing too much overhead. Further, even in the highly unlikely case of profiling failures (e.g., frequent changes in latency distributions), McTail can fall back to the default/baseline performance-aware GLB as a fail-over mechanism. Thus, McTail will not increase the energy cost compared to the default solution in any case.

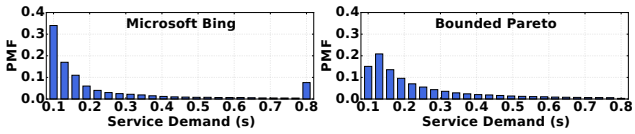


Fig. 4. Service demand distribution.

Example of performance profiling. We now show an example of performance profiling for the path between Chile and Shanghai for Microsoft Bing workload. We resort to a discrete-event simulator, Mathworks's SimEvents [29], that models request-level processing in geo-distributed systems. Details for the service time and network latency distributions are discussed in Section IV. Fig. 3(a) shows the data center latency distribution (queueing delay plus service time), while Fig. 3(b) shows the end-to-end Chile-Shanghai latency distribution obtained by performing a convolution on the data center latency and network latency distributions according to (12).

IV. Performance Evaluation

We now present our evaluation results for McTail, which minimizes the total electricity cost while meeting the tail latency constraint. We first describe the default settings we use to simulate a geo-distributed interactive service (Section IV-A), and then present our simulation results illustrating the performance of McTail (Section IV-B). We then discuss the impact of several factors, such as SLA and prediction errors, on McTail's performance (Sections IV-C and IV-D). Throughout the evaluation, we use Mathworks SimEvents for our large-scale simulations as discussed in Section III-C.

A. Settings

Simulator. SimEvents takes as inputs the service time and network latency distributions, which it then uses to simulate queueing and request processing. It is a popular discrete-event simulator and can well capture the real-world service process [30]. By default, we consider Microsoft Bing search workload [30]. The left figure in Fig. 4 shows the detailed service demand distribution used in our simulation. Bing workload originally has a service time between 5 and 120 milliseconds [30], which is much lower than inter-continental network latency. Thus, to make it more comparable to the network latency, we scale the Bing service demand to 0.1 to 0.8 seconds, while keeping the same relative distribution. The spike at the end of the Bing distribution in Fig. 4 indicates a timeout response. We also use bounded Pareto distribution for sensitivity analysis.

Traffic source and data center locations. We simulate five front-end gateways/traffic sources and nine data centers located across the world in three different regions: North

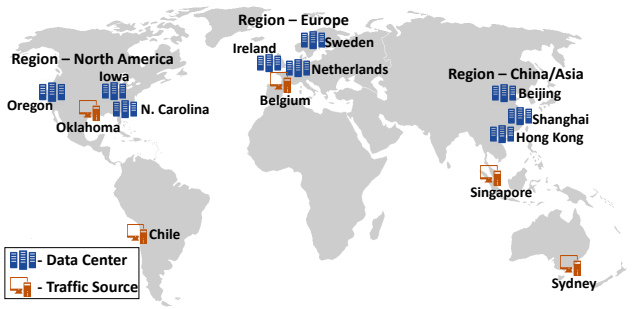


Fig. 5. Traffic source and data center locations used in our simulation setting.

America, Europe, and Asia (China). There are three data centers in each region. Although McTail does not require homogeneous data center settings, we use the same capacity at all data centers to simplify the simulation setting. The data center and request sources are shown in Fig. 5. The locations are chosen based on a subset of actual data center sites of Google and Facebook [31], [32]. For network latencies, we use half-normal distributions [33], where the mean and standard deviation depends on the distance between the source and the data center. We approximate the mean by considering round trip network latency of 1.64 milliseconds per 100 miles [34].

Workload trace. The source workload traces are taken from different services of Google and Microsoft [35], [36]. The trace data specifies the average normalized arrival rate over time and suffices for our purpose. The workloads are scaled to have a realistic average data center utilization of 30% [37].

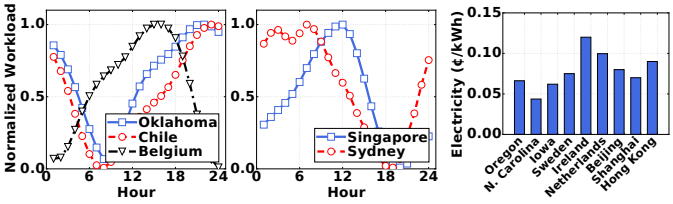
Energy costs. We use the electricity price at each data center location to determine the energy costs. For North America we use the local utility prices [38]–[40], for Europe we use the electricity prices reported in [41], and for China/Asia region we use [42]. The workload traces and electricity prices are shown in Fig. 6.

SLA and simulation settings. We set 1.5 seconds as the SLA threshold for p95 response time. The simulation period is 24 hours with load distribution decisions being updated every 15 minutes. We consider data center servers that have 40% static and 60% dynamic energy consumption.

Baseline. We compare McTail with a performance-aware but cost-oblivious approach widely used as a benchmark for GLB research [9]. This approach distributes the workload among all data center groups according to their capacities to balance the utilization. Since the data centers under consideration have the same capacity, the performance-aware will uniformly distribute workloads among data center groups. Hence, we call it EQL (EQual Load distribution). The existing GLB research [3], [4], [9], [10] does not apply to our problem setting and hence is not included for comparison.

B. Cost and performance

Fig. 7(a) shows the normalized cost of McTail and EQL, and Fig. 7(b) shows the performance in terms of probability that the response time meets the SLA latency threshold of 1.5 seconds. We see that McTail has lower energy costs than EQL throughout the simulation period since it exploits the difference in electricity prices in different data center locations.



(a) (b)

Fig. 6. Workload traces and electricity prices.

Over the entire 24 hours period, McTail saves more than 7% in electricity costs while ensuring that the probability of response time being less than 1.5 seconds remains above 95%. Note that, in Fig. 7(b), the latency threshold satisfaction probability for at least one of the source locations is very close to 95% at all times. This makes sense intuitively as otherwise McTail could realize greater cost savings by shifting additional workload to cheaper data center locations as long as the latency threshold satisfaction probability remains above 95% for all sources.

Fig. 7(c) shows how McTail takes advantage of cheaper electricity by sending more workloads to data centers with low electricity prices. As each request is simultaneously sent to all the regions for processing, the total amount of workload at each region is the same. Therefore, we only show the North America region for illustration. Observe that around time slot 48, McTail shifts workload from N. Carolina to the other two data centers as the electricity price of N. Carolina goes up. Later, at around time slot 80, workload is again shifted back to N. Carolina as its electricity price drops.

C. Impact of SLA

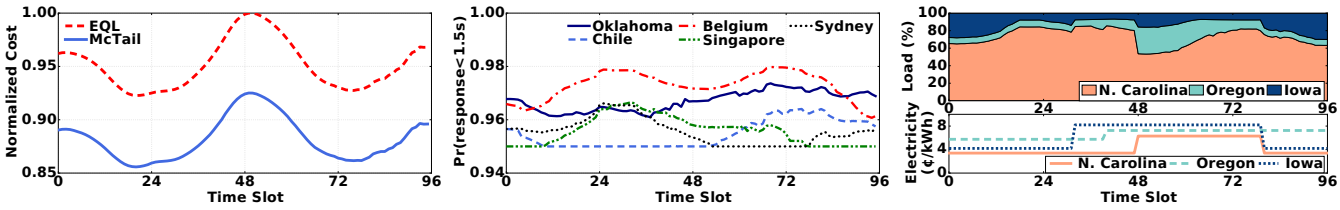
The SLA constraints can impact the performance and cost savings of McTail considerably. We study the impact of changing the two SLA parameters, namely, the response time threshold D_i, and the tail percentile P_i^{SLA}.

Fig. 8 shows our results for different SLA parameters. Keeping the tail percentile at 95, we vary the response time threshold from 1.5 seconds to 2 seconds, and show the cost savings in Fig. 8(a). The whiskers represent the maximum and minimum cost savings in any time slot over the 24 hour simulation period. The box represents the lower (25%) and upper quartile (75%) savings. The markers inside the box represent the mean. We see that when the response time threshold is increased (i.e., relaxed), McTail yields better cost savings. This is intuitive as with a relaxed threshold, McTail can process more work in economical data centers without violating the SLA.

Fig. 8(b) illustrates the impact of tail percentile setting. We keep the response time threshold fixed at 1.5 seconds and vary the tail percentile from 90 to 95. Again, as expected, a lower percentile provides greater flexibility for McTail, and results in increased cost savings.

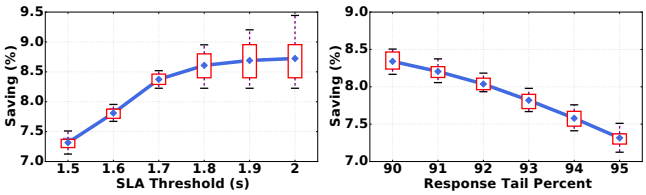
D. Sensitivity analysis

Response time profiling error. We use the response time distribution of each source to data center path in McTail.



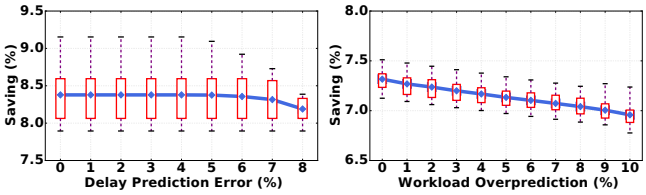
(a) (b) (c)

Fig. 7. Comparison of cost and performance between McTail and EQL. (a) Normalized electricity cost over time. (b) Probability that response time is below the 1.5 seconds SLA threshold. (c) Load distribution across the three data centers in North America.



(a) (b)

Fig. 8. Impact of change in response time threshold D_i and tail percentile SLA target P_i^{SLA}. Relaxing the SLA constraints increases the cost savings under McTail.

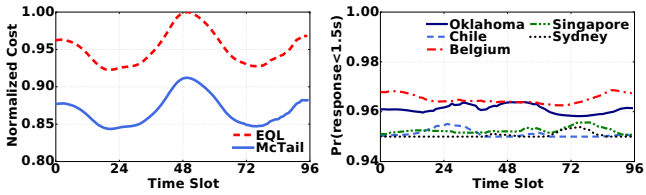


(a) (b)

Fig. 9. Impact of delay prediction error and conservative workload prediction.

However, there could be some error in the response time distribution profiling due to slow profiling or outliers. One conservative way to handle this uncertainty is to overpredict the response time to allow some room for errors, while still meeting the tail latency constraint. In Fig. 9(a), we study how overprediction affects McTail in terms of cost savings. Specifically, we underestimate the probability that the response time is less than our delay threshold along each path, by scaling down the profiled probability. Then, instead of considering the default p95 latency constraint, we relax the SLA to the p90 latency constraint (with the same threshold), since there may not be a feasible solution if we underestimate the latency satisfaction probability. We see that the cost savings largely remain unchanged even at 8% overprediction, demonstrating the robustness of McTail against latency profiling errors.

Workload prediction. Another important aspect of McTail is that it updates the load distribution matrix $\boldsymbol{\lambda}$ periodically (e.g., every 15 minutes) based on estimated workload for the next decision slot. Similar to response time, workload overprediction can be employed to keep head room for prediction errors. In Fig. 9(b), we study the impact of workload overprediction on cost savings. We overpredict the workload during decision making but use the actual workload when we determine the electricity costs. We see that although high overprediction decreases the cost savings, the reduction is very small, less than 0.5% for even a 10% overprediction. This shows that McTail is not very sensitive to workload prediction error.



(a) (b)

Fig. 10. Cost and performance with bounded Pareto distribution.

Service time distribution. We now consider a synthetic service time distribution — bounded Pareto distribution, between 0.1 to 0.8 seconds as illustrated in the right figure of Fig. 4, which is shown to be representative of real-world distributions [43]. Fig. 10 shows the cost and performance for this distribution. We see that the results are qualitatively similar to those in Figs. 7(a) and 7(b): McTail reduces the cost by roughly 8% compared to EQL while meeting the p95 latency threshold of 1.5 seconds. This demonstrates that McTail works well under different service time distributions.

V. Related Work

Data center energy management has become increasingly important as the electricity price continues to grow. For example, energy proportionality by dynamic server provisioning [7], [44] has been a proven technique to cut energy consumption. Our study is mostly related to energy management in large geo-distributed data center systems, for which many studies have exploited spatial diversities to optimize GLB while meeting latency constraints. Here, we briefly discuss these works. [3], [8]–[10] schedules workloads to data centers to lower electricity prices for cost savings, [5], [11] considers the on-site intermittent renewables and studies GLB techniques that "follow the renewables", and [4] considers the carbon efficiency diversity and routes workloads to greener data centers. These studies, however, all assume that each request is processed in only one data center, which does not apply to geo-distributed interactive services which need processing over multiple data centers simultaneously due to geo-distributed data that is costly and/or forbidden to move across different regions.

Geo-distributed workload processing has been quickly emerging as an important workload and has received much attention recently [16], [17], [23], [45]. For example, [45] exploits adaptive execution and trades accuracy for responsiveness, [16] proposes some heuristics to optimize data and task placement across geo-distributed systems, while [17] stud-

ies coordinated scheduling across data centers such that the scheduling inside one data center also considers the utilization and congestion level in other data centers. These solutions are complementary to our study, as they focus on scheduling workloads inside data centers while we focus on GLB across data centers. Further, these studies do not exploit the spatial diversity of electricity prices to minimize the total electricity cost while meeting a tail latency constraint.

VI. Conclusion

In this paper, we made an early effort to optimize GLB decisions that minimize the electricity cost for the emerging geo-distributed interactive services subject to a tail latency constraint. Compared to the rich literature on GLB, we made two contributions: first, we formulated the GLB problem for geo-distributed interactive services which rely on request processing in multiple data centers due to distributed data sets; second, we proposed an efficient algorithm, McTail, that employs a data-driven approach to efficiently determine the tail latency performance. Finally, we performed an event-based simulation study to validate McTail.

Acknowledgement

This work was supported in part by the U.S. National Science Foundation under grants CNS-1622832, CNS-1464151, CNS-1551661, CNS-1565474, and ECCS-1610471.

References

[1] Y. Sverdlik, "Google to build and lease data centers in big cloud expansion," in *DataCenterKnowledge*, April 2016.

[2] I. Goiri, W. Katsak, K. Le, T. D. Nguyen, and R. Bianchini, "Parasol and greenswitch: managing datacenters powered by renewable energy," in *ASPLOS*, 2013.

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

[4] P. X. Gao, A. R. Curtis, B. Wong, and S. Keshav, "It's not easy being green," *SIGCOMM Comput. Commun. Rev.*, 2012.

[5] K. Le, R. Bianchini, J. Zhang, Y. Jaluria, J. Meng, and T. D. Nguyen, "Reducing electricity cost through virtual machine placement in high performance computing clouds," in *SuperComputing*, 2011.

[6] C. Wang, B. Urgaonkar, Q. Wang, and G. Kesidis, "A hierarchical demand response framework for data center power cost optimization under real-world electricity pricing," in *MASCOTS*, 2014.

[7] R. Singh, D. Irwin, P. Shenoy, and K. K. Ramakrishnan, "Yank: Enabling green data centers to pull the plug," in *NSDI*, 2013.

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

[9] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Greening geographical load balancing," in *SIGMETRICS*, 2011.

[10] L. Rao, X. Liu, L. Xie, and W. Liu, "Reducing electricity cost: Optimization of distributed internet data centers in a multi-electricity-market environment," in *INFOCOM*, 2010.

[11] Y. Zhang, Y. Wang, and X. Wang, "Electricity bill capping for cloud-scale data centers that impact the power markets," in *ICPP*, 2012.

[12] J. Lin and A. Kolcz, "Large-scale machine learning at twitter," SIGMOD, 2012.

[13] Y. Zhang, Y. Wang, and X. Wang, "Greenware: Greening cloud-scale data centers to maximize the use of renewable energy," in *Middleware*, 2011.

[14] D. S. Palasamudram, R. K. Sitaraman, B. Urgaonkar, and R. Urgaonkar, "Using batteries to reduce the power costs of internet-scale distributed networks," in *SoCC*, 2012.

[15] A. Vulimiri, C. Curino, P. B. Godfrey, T. Jungblut, J. Padhye, and G. Varghese, "Global analytics in the face of bandwidth and regulatory constraints," in *NSDI*, 2015.

[16] Q. Pu, G. Ananthanarayanan, P. Bodik, S. Kandula, A. Akella, P. Bahl, and I. Stoica, "Low latency geo-distributed data analytics," SIGCOMM, 2015.

[17] C.-C. Hung, L. Golubchik, and M. Yu, "Scheduling jobs across geo-distributed datacenters," in *SoCC*, (New York, NY, USA), 2015.

[18] M. Rost and K. Bock, "Privacy by design and the new protection goals*," *DuD, January*, 2011.

[19] P. Mohan, A. Thakurta, E. Shi, D. Song, and D. Culler, "Gupt: Privacy preserving data analysis made easy," SIGMOD '12, 2012.

[20] Y. Chen, S. Alspaugh, D. Borthakur, and R. Katz, "Energy efficiency for large-scale mapreduce workloads with significant interactive analysis," in *EuroSys*, 2012.

[21] J.-M. Yun, Y. He, S. Elnikety, and S. Ren, "Optimal aggregation policy for reducing tail latency of web search," in *SIGIR*, 2015.

[22] M. E. Haque, Y. h. Eom, Y. He, S. Elnikety, R. Bianchini, and K. S. McKinley, "Few-to-many: Incremental parallelism for reducing tail latency in interactive services," in *ASPLOS*, 2015.

[23] F. Nawab, D. Agrawal, and A. E. Abbadi, "The challenges of global-scale data management," in *SIGMOD (Tutorial)*, 2016.

[24] X. Fan, W.-D. Weber, and L. A. Barroso, "Power provisioning for a warehouse-sized computer," in *ISCA*, 2007.

[25] Y. He, S. Elnikety, J. Larus, and C. Yan, "Zeta: Scheduling interactive services with partial execution," in *SOCC*, 2012.

[26] I. Goiri, K. Le, T. D. Nguyen, J. Guitart, J. Torres, and R. Bianchini, "Greenhadoop: leveraging green energy in data-processing frameworks," in *EuroSys*, 2012.

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

[28] A. Gandhi and J. Chan, "Analyzing the Network for AWS Distributed Cloud Computing," *SIGMETRICS Performance Evaluation Review*, vol. 43, no. 3, pp. 12–15, 2015.

[29] "SimEvents - Model and simulate discrete-event systems," http://www.mathworks.com/products/simevents/.

[30] S. Ren, Y. He, S. Elnikety, and K. S. McKinley, "Exploiting processor heterogeneity in interactive services," in *ICAC*, 2013.

[31] "Google - Data center locations," https://www.google.com/about/datacenters/inside/locations/index.html.

[32] "ZDNet - Facebook's data centers worldwide, by the numbers and in pictures," http://www.zdnet.com/pictures/facebooks-data-centers-worldwide-by-the-numbers-and-in-pictures/.

[33] G. Hooghiemstra and P. Van Mieghem, "Delay distributions on fixed internet paths," tech. rep., Delft University of Technology, 2001.

[34] "Cisco - Design Best Practices for Latency Optimization," https://www.cisco.com/application/pdf/en/us/guest/netsol/ns407/c654/ccmigration_09186a008091d542.pdf.

[35] "Google transparency report," http://www.google.com/transparencyreport/traffic/explorer.

[36] E. Thereska, A. Donnelly, and D. Narayanan, "Sierra: a power-proportional, distributed storage system," *Tech. Rep. MSR-TR-2009-153*, 2009.

[37] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33–37, 2007.

[38] "Portland General Electric," https://www.portlandgeneral.com/-/media/public/documents/rate-schedules/sched_083.pdf.

[39] "Duke Energy," http://www.duke-energy.com/pdfs/NCScheduleOPTV.pdf.

[40] "Waverly Light and Power," http://www.waverlyutilities.com/webres/File/Commercial/Gen%20&%20Municipal.pdf.

[41] "Electricity price statistics," http://ec.europa.eu/eurostat/statistics-explained/index.php/Electricity_price_statistics.

[42] "Average electricity prices around the world: $/kWh," https://www.ovoenergy.com/guides/energy-guides/average-electricity-prices-kwh.html.

[43] P. Barford and M. Crovella, "A performance evaluation of hyper text transfer protocols," in *SIGMETRICS*, 1999.

[44] A. Gandhi, M. Harchol-Balter, R. Raghunathan, and M. A. Kozuch, "Autoscale: Dynamic, robust capacity management for multi-tier data centers," *ACM Trans. Comput. Syst.*, vol. 30, pp. 14:1–14:26, Nov. 2012.

[45] B. Heintz, A. Chandra, and R. K. Sitaraman, "Trading timeliness and accuracy in geo-distributed streaming analytics," in *UMN Tech. Report 16-003*, 2016.