2012 32nd IEEE International Conference on Distributed Computing Systems

Provably-Efficient Job Scheduling for Energy and Fairness in Geographically Distributed Data Centers

Shaolei Ren
Electrical Engineering Department
University of California, Los Angeles
Los Angeles, CA 90095

Yuxiong He
Microsoft Research
One Microsoft Way
Redmond, WA 98052

Fei Xu
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

***Abstract*— Decreasing the soaring energy cost is imperative in large data centers. Meanwhile, limited computational resources need to be fairly allocated among different organizations. Latency is another major concern for resource management. Nevertheless, energy cost, resource allocation fairness, and latency are important but often contradicting metrics on scheduling data center workloads.**

In this paper, we explore the benefit of electricity price variations across time and locations. We study the problem of scheduling batch jobs, which originate from multiple organizations/users and are scheduled to multiple geographically-distributed data centers. We propose a provably-efficient online scheduling algorithm – GreFar – which optimizes the energy cost and fairness among different organizations subject to queueing delay constraints. GreFar does not require any statistical information of workload arrivals or electricity prices. We prove that it can minimize the cost (in terms of an affine combination of energy cost and weighted fairness) arbitrarily close to that of the optimal offline algorithm with future information. Moreover, by appropriately setting the control parameters, GreFar achieves a desirable tradeoff among energy cost, fairness and latency.

I. Introduction

With the emergence of cloud computing services, there has been a growing trend toward large-scale and geographically distributed data centers. Many megawatts of electricity are required to power such data centers, and companies like Google and Microsoft spend a large portion of their overall operational costs (e.g., tens of millions of dollars) on electricity bills [1].

Better energy efficiency of servers and lower electricity prices are both important in reducing the energy cost. Given the heterogeneity of servers in terms of energy efficiency and the diversity of electricity prices over geographically distributed data centers and over time, the key idea is to preferentially shift power draw (1) to energy-efficient servers and (2) to places and times offering cheaper electricity prices.

In addition to reducing energy cost, satisfying fairness and delay constraints is also important. For example, when multiple organizations or groups of users share the resources, allowing jobs from a few users to run first while deferring other jobs for lower electricity prices may result in reduced energy cost, but may sacrifice the response time of other users and adversely affect fairness among users. Since the performance metrics of energy, fairness and delay are often contradicting, it is desirable to have a tunable system with the flexibility to meet different business requirements.

In this paper, we consider the problem of developing an online scheduler that distributes batch workloads geographically across multiple data centers and to heterogeneous servers for minimizing energy cost with fairness consideration subject to delay requirements. This is a challenging problem because data centers experience time-varying workloads, server availability and electricity prices with possibly unknown statistics and/or non-stationary distributions. Even though some statistics may be estimated or predicted, applying traditional optimization techniques such as dynamic programming to compute the globally optimal solution can be time consuming and hence, it is not applicable for online schedulers in practice [4].

We propose a practical yet provably-efficient online scheduling algorithm "GreFar" to solve this problem, which is inspired by the recently developed technique of Lyapunov optimization [11] for time-varying systems. Our algorithm does not require any prior knowledge of the system statistics (which can even be non-stationary) or any prediction on future job arrivals and server availability. Moreover, it is computationally efficient and easy to implement in large practical systems. GreFar constructs and solves an oline optimal problem based on the current job queue lengths, server availability and electricity prices; the solution is proven to offer close to the offline optimal performance with future information. More precisely, given a cost-delay parameter $V \geq 0$, GreFar is $O(1/V)$-optimal with respect to the average (energy-fairness) cost against the offline optimal algorithm while bounding the queue length by $O(V)$. Without considering fairness, the energy-fairness cost solely represents the energy cost, while with fairness taken into account, it is an affine combination of energy cost and fairness score (which is obtained through a fairness function). Furthermore, our algorithm is associated with two control parameters, i.e., cost-delay parameter and energy-fairness parameter, which can be appropriately tuned to provide a desired performance tradeoff among energy, fairness and queueing delay.

To complement the analysis, we conduct a simulation study to evaluate our algorithm using workloads from Microsoft Cosmos clusters.[1] Our results show that: (1) GreFar effectively reduces the energy cost (at the expense of

[1]Cosmos is Microsoft's internal cloud computing infrastructure for storing and analyzing massive data sets. It is designed to run on large clusters consisting of thousands of commodity servers and provide highly reliable storage and highly scalable computation.

1063-6927/12 $26.00 © 2012 IEEE
DOI 10.1109/ICDCS.2012.77

increased delay) by opportunistically processing batch jobs using energy-efficient servers and when electricity prices are sufficiently low; (2) With an appropriate energy-fairness parameter, GreFar achieves much higher fairness while only incurring a marginal increase in energy cost; (3) GreFar is flexible in achieving a desirable tradeoff between energy cost, fairness and queueing delay.

The remainder of the paper is organized as follows. Section II discusses related work. Section III describes the system, job and scheduling models. Section IV and V presents the online algorithm and its analysis. Section VI shows the performance evaluation results. Finally, Section VII concludes the paper.

II. Related Work

There has been a growing interest in cutting electricity bills for large data centers. Several prior studies explore the opportunity of energy saving by executing jobs when and/or where the electricity prices are low (e.g., [5], [6], [7], [8], [9]). Among them, some perform local optimization at each time period without considering the electricity variations across time periods; thus, they do not offer performance guarantees for the average energy cost or queueing delay over a large time horizon (e.g., [5], [6]). Some prior studies assume that the electricity price variations and/or job arrivals follow certain stationary (although possibly unknown) distributions ([7], [8], [9]). In practice, the job arrivals may not follow any stationary distributions, especially in an enterprise computing environment where different organizations only submit job requests sporadically (see Fig. 1 for a three-day trace of Microsoft Cosmos workloads).

There are other related studies on scheduling workloads across multiple data centers. Lin et al. [3] propose an online right-sizing algorithm which dynamically turns on/off servers to minimizes the delay plus energy cost, under the assumption that the electricity price is fixed over time. Guenter et al. [4] consider a similar problem but proposes to predict the future service demand using a Markov chain to determine the number of active servers. Later, Buchbinder et al. [2] study the problem by taking the bandwidth cost into consideration; the proposed solution does not address the queueing delay requirement. Qureshi et al. [1] quantify the economic gains by scheduling workloads across multiple data centers, which is an empirical study without providing analytical performance bounds on the proposed scheduling algorithm.

Compared with the existing research, GreFar does not require any prior knowledge or assume any (stationary) distributions of the system dynamics. It minimizes the average energy-fairness cost while bounding the average queueing delay based on purely online information. Moreover, it addresses fairness, an important factor in many shared large-scale systems such as Microsoft Cosmos. To our best knowledge, GreFar is the first provably-efficient solution that minimizes the energy-fairness cost with queueing delay guarantees under arbitrarily time-varying system dynamics (e.g., non-stationary random server availability, electricity prices, job arrivals).

III. Problem Formulation

This section describes the data center system model, job model and our scheduling model. We consider a system with time indices $t = \{0, 1, 2, \cdots\}$, where each t represents a scheduling instant.

A. Data Center System Model

There are N geographically distributed data centers, each of which houses thousands of servers. Servers may be homogeneous or heterogeneous in hardware and performance characteristics. One major source of heterogeneity is that data centers operate several generations of servers from multiple vendors. Application needs, hardware innovations and prices jointly determine which type of servers to purchase. Our model accommodates both homogeneous and heterogeneous servers and, without loss of generality, we consider that there are $K \geq 1$ types of servers. Each type-k server is characterized by three parameters: processing speed s_k, idle power $\tilde{p}_k$ and active power p_k, where $p_k > \tilde{p}_k$.[2]

Next, we specify the *state* of each data center, which is time-varying and captures the randomness in the environment.

1) Server availability: Server availability may change over time due to different reasons server failures, software upgrades, influence of other workloads, etc. For example, the increase of interactive workloads may reduce the number of servers available to process batch jobs. We model the time-varying server availability using $n_{i,k}(t)$, which denotes the number of type-k servers that are available for processing batch jobs in data center i during the time t. To make the notation more concise, we use the vectorial expression $\mathbf{n}_i(t) = [n_{i,1}(t), n_{i,2}(t), \cdots, n_{i,K}(t)]$.

2) Electricity price: Due to the deregulation of electricity markets, electricity prices stochastically vary over time (e.g., every hour or 15 minutes) and across different locations [7][13][14]. We use $\varphi_i(t)$ to denote the electricity price in data center i during the time t. The function $\varphi_i(t)$ maps the total energy usage in data center i during time t to the electricity price. For the ease of presentation, we consider a simple example of $\varphi_i(t)$ which is a constant electricity price regardless of the actual energy usage during time t. The constant electricity price during a time period has been widely studied in prior works (e.g., [3]). However, our model and analysis are still applicable when the total electricity cost is not a linear function of the energy consumption. For example, the electricity cost can be an increasing and convex (or other) function of the energy consumption [2]. In such scenarios, the amount of other workloads such as interactive workloads also affects the energy price, and hence, we need to add another component, i.e., energy consumed by other workloads during each time t, into the data center state.

[2]Other server states, such as "low performance", can also be captured if we include the server state selection into the scheduling decisions (detailed in Section III-C).

Based on the above discussion, we can mathematically represent the state of data center i during time t using a tuple $\mathbf{x}_i(t) = \{\mathbf{n}_i(t), \varphi_i(t)\}$, for $i = 1, 2, \cdots, N$. Throughout the paper, we also use the vectorial expression $\mathbf{x}(t) = [\mathbf{x}_1(t), \mathbf{x}_2(t), \cdots, \mathbf{x}_N(t)]$ wherever applicable. As we have noted, $\mathbf{x}_i(t)$ is stochastically changing over time and the actual value of $\mathbf{x}_i(t)$ is not revealed until the beginning of time t. Unlike in prior works [7]–[9], we do not impose any restriction on the distribution of $\mathbf{x}_i(t)$, such as independent and identically distributed (i.i.d.).

B. Job Model

Each job, or service request, is characterized by a tuple $\{d, \mathcal{D}, \rho\}$. We use $d > 0$ to represent the service demand (i.e., job length) in terms of processor cycles; $\mathcal{D} \in \{1, 2, \cdots, N\}$ represent the set of data centers that this job can be scheduled to, which often relates to where the job's data is stored; $\rho \in \{1, 2, \cdots, M\}$ (where M is the total number of accounts) is the account from which the job originates, where an account may represent a user, a group of users or an organization.[3] During its execution, a job can be suspended and resumed later.

Depending on the job characteristics, we classify the jobs into $J \geq 1$ types and refer to the jobs belonging to type j as type-j jobs, for $j = 1, 2, \cdots, J$. A type-j job can be fully specified using $\mathbf{y}_j = \{d_j, \mathcal{D}_j, \rho_j\}$. In practice, we can group jobs having approximately the same characteristics into the same type. We denote by $a_j(t)$ the number of arrival jobs of type j during time slot t. The (time-varying) arrival process $\{a_j(t) \in \mathbb{Z}^+,\ j = 1, 2, \cdots, J,\ \text{and}\ t = 0, 1, 2, \cdots\}$ does not necessarily follow any stationary distributions and can be arbitrary, while the only assumption, which is not a limiting factor in practice, is boundedness:

$$0 \leq a_j(t) \leq a_j^{\max}, \tag{1}$$

for a finite positive integer $a_j^{\max}$, $j = 1, 2, \cdots, J$ and $t = 0, 1, 2, \cdots$.

In the model, we assume that jobs are fully parallelizable. In other words, a job can be processed by any number of servers simultaneously. In practice, however, it may be possible that only a certain number of servers can process a job in parallel. Our model can be adapted easily to capture this fact by adding a parallelism constraint for each job type. Specifically, we need to add a constraint on the scheduling decisions (detailed in the next subsection) such that the maximum number of servers that can be used to process a job simultaneously is upper bounded.

C. Scheduling Model

1) Performance Metrics: We consider three important but often contradicting performance metrics in practice: energy cost, fairness and queueing delay.

[3]In this study, we neglect the jobs' service demands in terms of memory, storage, etc., which can be considered if we extend the service demand $d > 0$ from a scalar to a vector in which each element corresponds to one type of demand.

Energy cost: Energy consumption is one of the largest contributing factor to data center cost. Our scheduler considers time-varying electricity prices across data centers to decide where and when to run jobs to reduce the energy cost. We consider turning servers on/off as external events which affect the availability of servers but are orthogonal to our scheduling decisions; given the available servers, our scheduler decides which servers to use and which to stay idle under power saving mode. Thus, what matters is the difference in power consumption of servers between "busy" and "idle" states. Without loss of generality, we let $\tilde{p}_i = 0$ and hence, p_i represents the energy consumption of a busy type-i server minus that of an idle type-i server.

When $0 \leq b_{i,k}(t) \leq n_{i,k}(t)$ type-k servers in data center i are busy processing batch jobs, the energy cost for processing the scheduled batch jobs in data center i during time t is[4]

$$e_i(t) = \varphi_i(t) \cdot \sum_{k=1}^{K} b_{i,k}(t) p_k. \tag{2}$$

Thus, the total energy cost during time t is $e(t) = \sum_{i=1}^{N} e_i(t)$.

Fairness: Fairness in resource allocation among different accounts is an important concern for data centers. We define a *fairness* function to mathematically characterize the fairness of resource allocation. There are various fairness functions such as α-fair index [12]. In this paper, we use the following fairness function[5]

$$f(t) = -\sum_{m=1}^{M} \big[\frac{r_m(t)}{R(t)} - \gamma_m\big]^2, \tag{3}$$

where $r_m(t)$ is the total amount of computing resource allocated to all the jobs from account m during time t, $R(t) = \sum_{i=1}^{N} \sum_{k=1}^{K} n_{i,k}(t) s_k$ is the total available computing resource during time t, and $\gamma_m \geq 0$ is the weighting parameter indicating the desire amount of resource allocation to account m. The fairness score is maximized when $r_m(t) = \gamma_m R(t)$ for $m = 1, 2, \cdots, M$.

Queueing delay: Given a job arrival process, queueing delay is closely related to the average number of jobs in the queue. In this paper, we bound the queue length, which in turn determines the average delay performance. For each type of jobs, there are separate queues maintained at the central scheduler and in each data center. For type-j jobs, we denote the queue lengths at time t at the central scheduler and in data center i by $Q_j(t)$ and $q_{i,j}(t)$, respectively.

2) Scheduler Decisions: Control decisions, including job scheduling and server allocation, are made at the beginning of each time t. In practice, the duration of each time t depends on the scheduling quantum. For example, we may consider 15 minutes or 1 hour as the duration of a time slot, which correspond to the price updating frequencies in

[4]Each time t has the same duration and hence, we normalize the duration to one and the duration does not appear in the energy cost expression (2).

[5]Our analysis also applies if other fairness functions are considered.

electricity markets [13][14]. At the beginning of time t, the following decisions are made.

(1) $r_{i,j}(t) \in \mathbb{Z}^+$: the number of type-$j$ jobs scheduled to data center i during time t, for $i = 1, 2, \cdots, N$ and $j = 1, 2, \cdots, j$. A job cannot be split into multiple parts and hence the scheduling decision $r_{i,j}(t)$ takes integer values.

(2) $h_{i,j}(t) \in \mathbb{R}^+$: the number of type-$j$ jobs processed in data center i during time t, for $i = 1, 2, \cdots, N$ and $j = 1, 2, \cdots, J$. Since we allow a job to be paused during its execution, $h_{i,j}(t)$ is not necessarily an integer.

(3) $0 \leq b_{i,k}(t) \leq n_{i,k}(t)$: the number of type-$k$ servers in data center i during time t. Note that if a type-k server is turned on for a fraction of the time slot t, $b_{i,k}(t)$ may not be an integer. Moreover, $\sum_{j=1}^{J} h_{i,j}(t)d_j \leq \sum_{k=1}^{K} b_{i,k}(t)s_k$ is satisfied, i.e., the processed work cannot exceed the provided computing resource.

For notational convenience, we write the decisions, which we also refer to as *action*, made at the beginning of time t in a compact form as $\mathbf{z}(t) = \{r_{i,j}(t), h_{i,j}(t), b_{i,k}(t),\ i = 1, 2, \cdots, N,\ j = 1, 2, \cdots, J,\ k = 1, 2, \cdots, K\}$. Finally, we impose a mild assumption on the scheduling decisions that the following boundedness conditions are satisfied:

$$0 \leq r_{i,j}(t) \leq r_{i,j}^{\max}, \tag{4}$$

$$0 \leq h_{i,j}(t) \leq h_{i,j}^{\max}, \tag{5}$$

for some finite $r_{i,j}^{\max}$ and $h_{i,j}^{\max}$.

IV. Scheduling Algorithm

This section presents an offline optimal formulation of the problem and a provably efficient online algorithm "GreFar".

A. Problem formulation

To jointly consider the energy cost and fairness, we define the instantaneous energy-fairness cost function as follows

$$g(t) = e(t) - \beta \cdot f(t) = \sum_{i=1}^{N} e_i(t) - \beta \cdot f(t), \tag{6}$$

where $\beta \geq 0$ is a scaler (called energy-fairness parameter) that transfers the achieved fairness into energy cost saving, and $e_i(t)$ and $f(t)$ are given in (2) and (3), respectively. In special cases, when $\beta = 0$, the scheduler does not consider the fairness in resource allocation, whereas when $\beta \to \infty$, the scheduler does not consider the energy cost. Affine combination of two performance metrics is a common approach in multi-objective optimization (e.g., [3][5][17]). Moreover, β is equivalent to the corresponding Lagrangian multiplier if we formulate the fairness as a constraint [16].

Data centers operate over a large time horizon (e.g., more than ten years), and minimizing the time-average cost is crucial in cutting electricity bills. Let $\bar{g}$ be the time average cost of $g(t)$ under a particular control policy implemented over a sufficiently large but finite time horizon with t_{end} time slots:

$$\bar{g} \triangleq \frac{1}{t_{end}} \sum_{\tau=0}^{t_{end}-1} g(\tau). \tag{7}$$

Similarly, we define $\bar{a}_i \triangleq \frac{1}{t_{end}} \sum_{\tau=0}^{t_{end}-1} a_i(\tau)$, $\bar{r}_{i,j} \triangleq \frac{1}{t_{end}} \sum_{\tau=0}^{t_{end}-1} r_{i,j}(\tau)$, and $\bar{h}_{i,j} \triangleq \frac{1}{t_{end}} \sum_{\tau=0}^{t_{end}-1} h_{i,j}(\tau)$.

The problem of minimizing the cost can be formulated as follows:

$$\min_{\mathbf{z}(t), t=0,1,2,\cdots,t_{end}-1} \bar{g} \tag{8}$$

$$\text{s.t.,} \quad \bar{a}_j \leq \sum_{i \in \mathcal{D}_j} \bar{r}_{i,j}, \forall j = 1, 2, \cdots, J, \tag{9}$$

$$\bar{r}_{i,j} \leq \bar{h}_{i,j}, \forall i \in \mathcal{D}_j, j = 1, 2, \cdots, J, \tag{10}$$

$$\sum_{j=1}^{J} h_{i,j}(t)d_j \leq \sum_{k=1}^{K} b_{i,k}(t)s_k \leq \sum_{k=1}^{K} n_{i,k}(t)s_k \tag{11}$$

where $\sum_{k=1}^{K} n_{i,k}(t)s_k$ in (11) is the maximum amount of work that can be processed in data center i during time t and specifies that the scheduling decisions $h_{i,j}(t)$, for $j = 1, 2, \cdots, J$, are bounded in a convex polyhedron.

To solve the optimization problem (8)–(11), we need offline information (e.g., future job arrivals) which is unavailable in practice. Thus, we develop an online algorithm that makes scheduling decisions based on the currently available information only.

B. Online Algorithm

Based on the recently developed Lyapunov optimization technique [11], this section presents an online algorithm "GreFar", whose performance is provably "good" compared to that of the optimal offline policy with T-step lookahead information. The intuition of GreFar is to trade the delay for energy-fairness cost saving by using the queue length as a guidance for making scheduling decisions: jobs are processed only when the queue length becomes sufficiently large and/or electricity prices are sufficiently low.

Before presenting the algorithm, we need to introduce "queue dynamics", which specifies the queue length changes governed by the scheduling decisions (and job arrivals). The queue dynamics is instrumental for the scheduler to make online decisions. We express the queue dynamics governed the action $\mathbf{z}(t) = \{r_{i,j}(t), h_{i,j}(t), b_{i,k}(t),\ i = 1, 2, \cdots, N,\ j = 1, 2, \cdots, J,\ k = 1, 2, \cdots, K\}$ as below:

$$Q_j(t+1) = \max\left[Q_j(t) - \sum_{i=1}^{N} r_{i,j}(t), 0\right] + a_j(t), \tag{12}$$

$$q_{i,j}(t+1) = \max\left[q_{i,j}(t) - h_{i,j}(t), 0\right] + r_{i,j}(t), \tag{13}$$

where $Q_j(t)$ is the queue length for type-j jobs at the central scheduler and $q_{i,j}(t)$ is the queue length for type-j jobs in data center i during time t.

We describe GreFar in Algorithm 1, which is purely online and requires only the current data center state and queue lengths as the input. Solving (14) subject to the constraint (11) is a convex optimization problem, to which efficient numerical algorithms (e.g., interior point method) exist [16]. In particular, if the fairness is not taken into account (i.e., $\beta = 0$ in (6)), solving (14) becomes a standard linear programming problem.

The parameter $V \geq 0$ is a control variable which we refer to as cost-delay parameter, and it can be tuned to different

Algorithm 1 GreFar Scheduling Algorithm

1: At the beginning of every time slot t, observe the data center state $\mathbf{x}(t)$ and the vector of current queue states $\Theta(t)$

2: Choose $r_{i,j}(t) \geq 0$ and $h_{i,j} \geq 0$ subject to (4)(5)(11) to minimize

$$V \cdot g(t) - \sum_{j=1}^{J} Q_j(t) \cdot \left[\sum_{i \in \mathcal{D}_j} r_{i,j}(t) \right] + \sum_{j=1}^{J} \sum_{i \in \mathcal{D}_j} q_{i,j}(t) \cdot \left[r_{i,j}(t) - h_{i,j}(t) \right], \quad (14)$$

where the cost $g(t)$ is defined in (6).

3: Update $Q_j(t)$ and $q_{i,j}(t)$ according to (12) and (13), respectively.

values to trade the queueing delay for the energy-fairness cost. For simplicity, let us consider $\beta = 0$ and explain the role of V in making scheduling decisions. By substituting (6) into (14), we see that the scheduler chooses $h_{i,j}(t) > 0$ only when the electricity price $\varphi_i(t)$ is sufficiently low such that $V \cdot (W\varphi_i(t)d_j)$ is smaller than the current queue length $q_{i,j}(t)$, where W is a positive constant that only depends on the server speeds and power consumptions. When V is larger, the scheduler will wait longer until the electricity price is sufficiently low relative to the queue length before scheduling the jobs for processing (i.e., opportunistically take the advantage of low electricity prices). As a result, the energy cost reduces at the expense of the increased delay. On the other hand, when V is smaller, the scheduler will schedule jobs even though the electricity price is not sufficiently low, resulting in an increased energy cost but reduced delay. When the fairness is taken into account in scheduling (i.e., $\beta > 0$), the role of V is similar and can achieve a tradeoff between the energy-fairness cost and delay.

V. Algorithm Analysis

This section shows that the proposed online algorithm is provably-efficient against an optimal algorithm with T-step of look-ahead information. This section first describes the T-step look-ahead policy, and then analyzes the performance of GreFar against it. More specifically, we show that, given a cost-delay parameter V, our algorithm is $O(1/V)$-optimal with respect to average cost against the optimal T-step lookahead policy, while the queue length is bounded by $O(V)$.

A. T-Step Lookahead Policy

Here, we present the T-step lookahead policy, which has full knowledge of the data center states and job arrivals in the next (up to) T time steps. If T is sufficiently large (e.g., in the extreme case $T = t_{end}$), the T-step lookahead policy also "approximately" (or exactly if $T = t_{end}$) minimizes the average cost in (8).

We divide the time horizon of t_{end} time steps into $R \in \mathbb{Z}^+$ frames, each of which contains T time steps such that $t_{end} = RT$. In the T-step lookahead algorithm, the scheduler has future information (i.e., data center states and job arrivals) up to the next T time steps and minimizes the cost subject to certain constraints. Specifically, the cost minimization problem over the r-th frame, for $r = 0, 1, \cdots, R-1$, can be formulated as

$$\min_{\mathbf{z}(t), t=rT, rT+1, \cdots, rT+T-1} \frac{1}{T} \sum_{t=rT}^{t=rT+T-1} g(t) \quad (15)$$

$$\text{s.t.,} \quad \sum_{t=rT}^{t=rT+T-1} \left(a_j(t) - \sum_{i \in \mathcal{D}_j} r_{i,j}(t) \right) \leq 0, \forall j, \quad (16)$$

$$\sum_{t=rT}^{t=rT+T-1} \left(r_{i,j}(t) - h_{i,j}(t) \right) \leq 0, \forall i \in \mathcal{D}_j, \forall j \quad (17)$$

$$\sum_{j=1}^{J} h_{i,j}(t) d_j \leq \sum_{k=1}^{K} b_{i,k}(t) s_k \leq \sum_{k=1}^{K} n_{i,k}(t) s_k \quad (18)$$

In the problem (15)–(18), we denote the infimum of $\frac{1}{T} \sum_{t=rT}^{t=rT+T-1} g(t)$ by G_r^*, which is achievable over the r-th frame considering all the actions including those that are chosen with the perfect knowledge of data center states and job arrivals over the entire frame. Thus, the minimum cost over R frames achieved by the optimal T-step lookahead policy is

$$\frac{1}{R} \sum_{r=0}^{R-1} G_r^* \ . \quad (19)$$

We shall show that our online algorithm can achieve a cost close to the value of (19).

B. Online Algorithm Analysis

This section presents the performance analysis of our proposed online algorithm compared with the optimal T-step lookahead policy.

Before showing the main theorem, we first present the slackness conditions, which bound the relationship between the resource demand and availability. These conditions are prerequisites of Theorem 1.[6]

Slackness Conditions: There exists a value $\delta > 0$ and a sequence of scheduling decisions $r_{i,j}(t)$ and $h_{i,j}(t)$ such that, for data center states $\mathbf{x}(t)$, $t = 0, 1, \cdots t_{end} - 1$, the following conditions are satisfied

$$a_j(t) \leq \sum_{i \in \mathcal{D}_j} r_{i,j}(t) - \delta, \ \forall j \quad (20)$$

$$r_{i,j}(t) \leq h_{i,j}(t) - \delta, \ \forall i \in \mathcal{D}_j, \forall j \quad (21)$$

$$\sum_{j}^{J} h_{i,j}(t) d_j \leq \sum_{k=1}^{K} n_{i,k}(t) s_k - \delta \ \forall i \in \mathcal{D}_j, \forall j. \quad (22)$$

[6]If we only assume that the problem (8)–(11) is *strongly* feasible without slackness conditions, the performance analysis of energy-fairness cost remains similar while the upper bound on the queue length may grow as the time passes [10].

The conditions (20) and (21) ensure that the scheduler can successfully schedule all arrived jobs with δ slackness, while the condition (22) ensures that the available computing resource is always enough to process all the scheduled jobs. In practice, these three conditions are mild and can be easily satisfied. In particularly, the computing resource in a data center is provisioned for the peak load, and thus the available computing resource is (almost) always sufficient for processing workloads, i.e., (22) holds in practice. In the worst case where the data center is overloaded, admission control techniques can be applied to complement our scheme.

Theorem 1 shows a cost bound and queue length bound for GreFar.

Theorem 1. Suppose that the boundedness conditions (1)(4)(5) and the slackness conditions (20)(21)(22) are satisfied for some $\delta > 0$, that the data center states $\mathbf{x}(t)$ and job arrivals $a_j(t)$ are arbitrarily random, for $t = 0, 1, \cdots t_{end} - 1$ and $j = 1, 2, \cdots J$, and that all the queue lengths are initially zero. Then, the following statements hold.

a. All queue lengths are bounded. For any time step $t = 0, 1, \cdots t_{end} - 1$, we have

$$Q_j(t), q_{i,j}(t) \leq \frac{VC_3}{\delta}, \tag{23}$$

where $V \geq 0$ and C_3 is a finite number defined in (39) in the appendix.

b. For any $T \in \mathbb{Z}^+$ and $R \in \mathbb{Z}^+$ such that $t_{end} = RT$, the energy-fairness cost achieved by GreFar satisfies

$$\bar{g}^* \leq \frac{1}{R} \sum_{r=0}^{R-1} G_r^* + \frac{B + D(T-1)}{V}, \tag{24}$$

where $\bar{g}^*$ is the cost achieved by GeoFar for the problem (8)–(11), B and D are (finite) constants defined in the appendix and G_r^* is the minimum cost in the r-th frame achieved by the T-step lookahead policy.

Proof: The proof is provided in the appendix. □

Theorem 1 shows that, given a cost-delay parameter V, our algorithm is $O(1/V)$-optimal with respect to the average energy-fairness cost against the optimal T-step lookahead policy, while the queue length is bounded by $O(V)$. More specifically, the inequality (23) bounds the queue length: the queue length (which is closely related to the average delay) is bounded by $O(V)$ where V is the cost-delay parameter in Algorithm 1. The queue length bound is tighter when V is smaller. The inequality (24) shows that the average energy-fairness cost is bounded within an additional $O(1/V)$ cost of that achieved by the optimal T-step lookahead policy. The cost bound is closer to the minimum cost when V is larger.

Theorem 1 also shows that, by appropriately tuning the cost-delay parameter V, we can achieve a desired tradeoff between the cost and queue lengths. In particular, by increasing the value of $V \geq 0$, the (energy-fairness) cost becomes closer to that achieved by the optimal T-step lookahead policy, whereas the upper bounds on the queue lengths become larger.

Table I
SERVER CONFIGURATION AND ELECTRICITY PRICE IN DATA CENTERS

DC	Speed	Power	Avg. Price	Avg. Energy Cost per Unit Work
#1	1.00	1.00	0.392	0.392
#2	0.75	0.60	0.433	0.346
#3	1.15	1.20	0.548	0.572

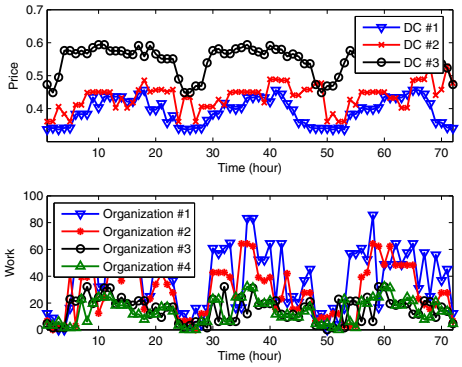


Figure 1. Three-day trace of electricity prices and total work of arrived jobs.

VI. PERFORMANCE EVALUATION

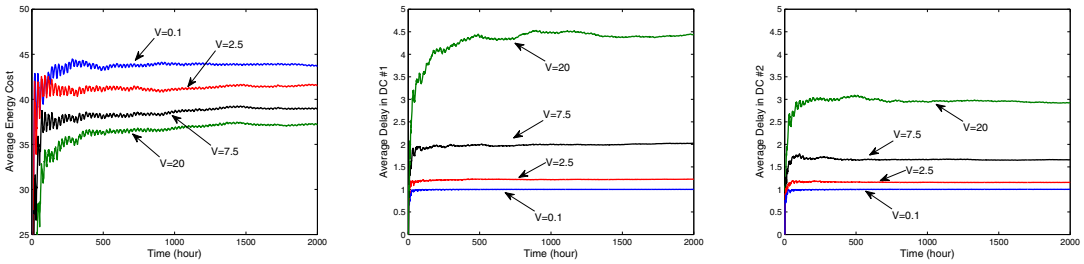
We perform a simulation study to evaluate our algorithm using workload distributions from Microsoft Cosmos clusters, which are Microsoft's internal cloud computing infrastructure consisting of thousands of commodity servers at three geographically distributed data centers. We conduct three sets of experiments:

- Minimizing energy cost with different V: study the impact of the cost-delay parameter V on energy cost minimization.
- Impact of the energy-fairness parameter β: study how β value affects the energy cost and fairness score.
- Algorithm comparison: compare GreFar with an online algorithm that always schedules jobs immediately whenever there are resources available.

The experimental results show that (1) GreFar reduces energy cost by opportunistically scheduling jobs when the electricity prices are sufficiently low and to servers with high energy efficiency; (2) With an appropriate energy-fairness parameter β, GreFar significantly increases the resource allocation fairness while only incurring a marginal increase in energy cost; (3) GeoFar is flexible in achieving a balance among energy cost, fairness and queueing delay.

A. Setup

We build a time-based simulator and drive the simulation using a real-world trace from Microsoft Cosmos clusters, where the trace includes the job arrival time, total work, organizational account information, data centers where the job is allowed to run, etc. To protect commercial business interests, we do not disclose the data center locations, the server quantity or hardware configurations. Instead, we



(a) Energy cost (b) Delay in DC #1 (c) Delay in DC #2

Figure 2. GreFar: minimize energy cost without fairness consideration (i.e., $\beta = 0$).

normalize the server speed and power consumption and scale the work of jobs accordingly.

We consider three data centers and four organizations. For illustration purposes, Table I only specifies one type of servers for each data center using normalized values, whereas each data center may have multiple types of servers in practice. We choose publicly available (hourly) electricity prices from [14] in locations with proximity to our considered data centers.[7] Fig. 1 shows a three-day trace of electricity prices. The fairness weighting parameters of the four organizations are 40%, 30%, 15% and 15%, respectively, indicating the desired resource allocation ratios. In the experiment, we scale the service demand: service demand 1 refers to 1000 hours on a server with a normalized speed of 1. Fig. 1 shows the total work of the arrived jobs over a three-day period from four organizations, reflecting the fact that the job arrivals are highly time dependent (e.g., more jobs during the day) and do not follow any stationary distributions. The (random) server availability is chosen such that it satisfies the slackness conditions (20)–(22).

B. Experimental Results

We discusses three sets of experimental results.

1) Minimizing energy cost with different V: Here, we study the impact of the cost-delay parameter V on the energy cost and delay of jobs. This experiment does not consider fairness requirement (i.e., $\beta = 0$). We show in Fig. 2 the average energy cost, and average delays in data center 1 and 2.[8] The average delays in data center 3 and at the central scheduler are similar and hence, we omit them due to space limitations.[9] The results conform with our analysis that with a greater V, GreFar is more efficient in cost minimization at the expense of increased queueing delays. The reason is that, with a large value of V, the scheduler will only schedule the jobs for processing when the electricity prices are sufficiently low. Thus, the low electricity prices are "opportunistically" utilized while high electricity prices are avoided, resulting in a reduced energy cost. On the other hand, when V is small (e.g., $V = 0.1$), the scheduler will schedule jobs even when the prices are not quite low, and thus the energy cost increases whereas the average delay decreases. Different settings of the cost-delay parameter V allows the system to operate subject to various business requirements on energy cost and queueing delay.

In addition to opportunistically processing jobs when the electricity prices are low, GreFar also schedules more arrived jobs to data centers that are more energy cost efficient, where the energy cost efficiency is related to both energy efficiency of data center hardware and the electricity price. For example, when $V = 7.5$ and $\beta = 100$, our simulation result shows that the average work per time step scheduled to data centers #1, #2, and #3 are 33.967, 48.502 and 14.770, respectively. In other words, more work is processed in data centers that incur lower energy costs (see Table I for the average energy cost per unit work).

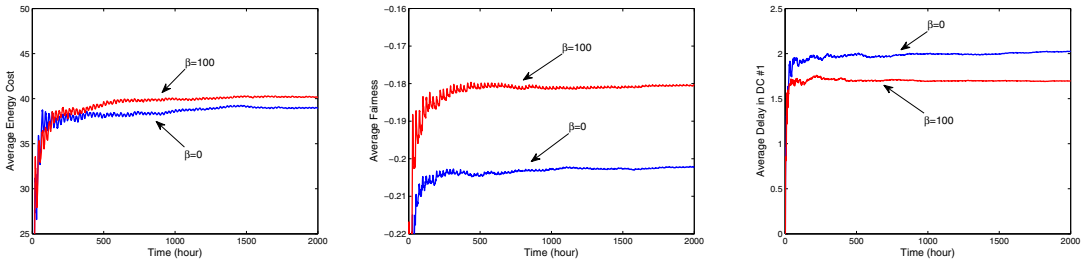
2) Impact of the energy-fairness parameter β: Fig. 3 compares the total energy cost, fairness, and delay in data center 1 with $\beta = 0$ and $\beta = 100$. According to our fairness function in (3), the highest (ideal) fairness score is 0 where all the resources are allocated and utilized based on the specified fairness weights. The higher the fairness score, the better fairness. Clearly, with $\beta = 100$, the fairness score is much higher than the score with $\beta = 0$. Nevertheless, the corresponding energy cost only increases marginally. Another positive impact is that the average delay also reduces, since this fairness model (Eqn. 3) has a side effect on encouraging the use of resources. For example, if all the servers are idle (i.e., $r_m(t) = 0$ for $m = 1, 2, \cdots, M$), the fairness score tends to be low. Therefore, with $\beta = 100$, GreFar schedules some jobs for processing even when the electricity prices are not very low to compensate the fairness score, which reduces the average queueing delay.

3) Algorithm comparison: We compare GreFar with another scheduling algorithm "Always", which always schedules the jobs immediately whenever there are resources

[7] If the electricity prices change every 15 minutes in real-time markets, our analysis still applies and the scheduler shall make scheduling decisions every 15 minutes.

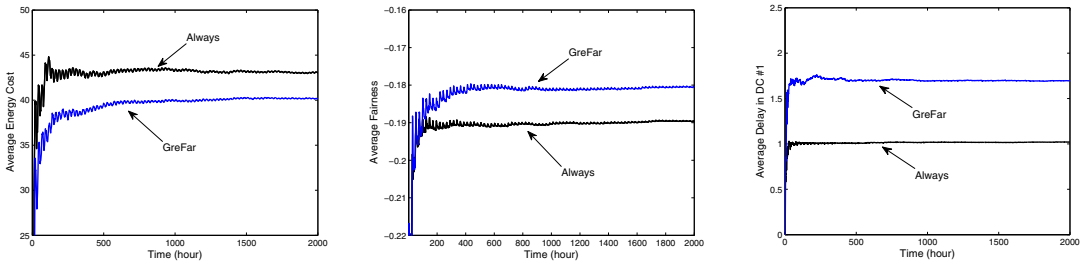
[8] Throughout this section, the average values at time $t = 1, 2, \cdots$ are obtained by summing up all the values up to time t and then dividing the sum by t.

[9] In Fig. 3 and Fig. 4, we only show the average delay in data center #1 for brevity.



(a) Energy cost (b) Fairness (c) Delay in DC #1

Figure 3. GreFar: minimize energy cost with fairness consideration.



(a) Energy cost (b) Fairness (c) Delay in DC #1

Figure 4. GreFar versus "Always" with $\beta = 100$ and $V = 7.5$.

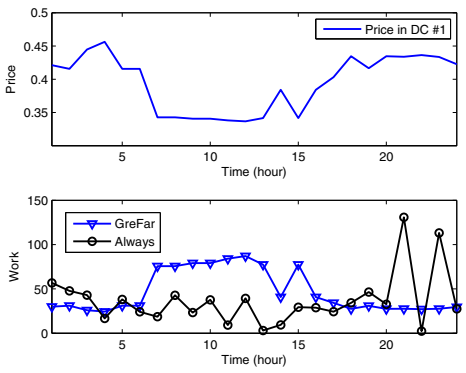


Figure 5. Scheduled work to process ($\beta = 0$ and $V = 7.5$).

available.[10] Since Always tries to schedule the jobs based on the server availability, most of the jobs will be scheduled in the next time slot upon their arrivals. Thus, the average delay is expected to be one. The result in Fig. 4 show that GreFar (with $V = 7.5$ and $\beta = 100$) incurs a lower energy cost and better fairness than Always at the expense of increased average delay.

We illustrate in Fig. 5 why GreFar is more efficient in energy cost minimization than Always. Fig. 5 shows a snapshot of one-day schedule in data center 1. "Always" schedules the jobs without taking into account the electricity prices. In contrast, GreFar can effectively avoid high electricity prices and schedule the jobs for processing when electricity prices are low, thereby reducing the energy cost.

[10]We do not show the performance of the optimal T-step lookahead algorithm, as GreFar is guaranteed to achieve a cost within $O(1/V)$ of that of the optimal T-step lookahead algorithm.

VII. Conclusion

This paper presents a provably-efficient online algorithm, GreFar, for scheduling batch jobs among multiple geographically distributed data centers. For arbitrarily random job arrivals, server availability and electricity prices, GreFar minimizes the energy-fairness cost while providing queueing delay guarantees. It opportunistically schedules jobs when electricity prices are sufficiently low and to places where the energy cost per unit work is low. Given the cost-delay parameter V, GreFar achieves a cost within $O(1/V)$ of the optimal T-step lookahead algorithm, while bounding the queue length by $O(V)$.

References

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

[2] N. Buchbinder, N. Jain, and I. Menache, "Online job migration for reducing the electricity bill in the cloud," *IFIP Networking*, 2011.

[3] M. Lin, A. Wierman, L. L. H. Andrew, and E. Thereska, "Dynamic right-sizing for power-proportional data centers," *Infocom*, 2011.

[4] B. Guenter, N. Jain, and C. Williams, "Managing cost, performance and reliability tradeoffs for energy-aware server provisioning," *Infocom*, 2011.

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

[6] L. Rao, X. Liu, L. Xie, and Wenyu Liu, "Reducing electricity cost: optimization of distributed Internet data centers in a multi-electricity-market environment," *Infocom*, 2010.

[7] D. Xu and X. Liu, "Geographic trough filling for Internet datacenters," *http://arxiv.org/abs/1108.5494*.

[8] Y. Yao, L. Huang, A. Sharma, L. Golubchik, and M. J. Neely, "Data centers power reduction: A two time scale approach for delay tolerant workloads," *Infocom*, 2012.

[9] Y. Guo, Z. Ding, Y. Fang, and D. Wu, "Cutting down electricity cost in Internet data centers by using energy storage," *Globecom*, 2011.

[10] M. J. Neely, "Universal scheduling for networks with arbitrary traffic, channels, and mobility," *Technical Report*, 2010.

[11] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Foundations and Trends in Networking*, vol. 1, no. 1, pp. 1-149,2006.

[12] J. W. Lee, M. Chiang, and R. A. Calderbank, "Utility-optimal random-access control," *IEEE Trans Wireless Commun.*, vol. 6, no. 7, pp. 2741-2751, 2007.

[13] California ISO, http://www.caiso.com

[14] Federal Energy Regulatory Commission, http://www.ferc.gov

[15] A. Leon-Garcia, *Probability, Statistics, and Random Processes for Electrical Engineering*, (3rd Ed.) Prentice Hall, 2008.

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

[17] D. P. Bertsekas, *Nonlinear Programming*, Athena Scientific, Belmont, MA, 1995.

APPENDIX

Proof: Here, we provide the proof of Theorem 1. First, we define the vector of all queue states during time t as

$$\Theta(t) \triangleq \{Q_j(t), q_{i,j},\ \forall i \in \mathcal{D}_j, \forall j = 1, 2, \cdots, J\}, \tag{25}$$

where $Q_j(t)$ is the queue length for type-j jobs at the central scheduler and $q_{i,j}(t)$ is the queue length for type-j jobs in data center i during time t. As a scalar measure of all the queue lengths, we define the quadratic Lyapunov function as

$$L(\Theta(t)) \triangleq \frac{1}{2}\sum_{j=1}^{J} Q_j^2(t) + \frac{1}{2}\sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}^2(t). \tag{26}$$

Let $\triangle_T(t)$ be the T-step Lyapunov drift yielded by some control policies over the interval $t, t+1, \cdots, t+T-1$:

$$\triangle_T(t) \triangleq L(\Theta(t+T)) - L(\Theta(t)). \tag{27}$$

Similarly, the 1-step drift is

$$\triangle_1(t) \triangleq L(\Theta(t+1)) - L(\Theta(t)). \tag{28}$$

Then, it can be shown that the 1-step drift satisfies

$$\begin{aligned}\triangle_1(t) \leq & B + \sum_{j=1}^{J} Q_j(t)\cdot[a_j(t) - \sum_{i\in\mathcal{D}_j} r_{i,j}(t)] \\ & + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}(t)\cdot[r_{i,j}(t) - h_{i,j}(t)],\end{aligned} \tag{29}$$

where B is a constant satisfying, for all $t = 0, 1, \cdots, t_{end}$,

$$B \geq \frac{1}{2}\big[a_j(t) + \sum_{i\in\mathcal{D}_j} r_{i,j}(t)\big] + \frac{1}{2}\left[r_{i,j}(t) + h_{i,j}(t)\right]^2, \tag{30}$$

where is finite due to the boundedness conditions (1)(4)(5).

Part (a): Based on (29), we can easily show that

$$\begin{aligned}\triangle_1(t) + V\cdot g(t) \leq & B + V\cdot g(t) + \sum_{j=1}^{J} Q_j(t)\cdot[a_j(t) - \sum_{i\in\mathcal{D}_j} r_{i,j}(t)] \\ & + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}(t)\cdot[r_{i,j}(t) - h_{i,j}(t)].\end{aligned} \tag{31}$$

Thus, GreFar actually minimizes the upper bound on the 1-step Lyapunov drift plus a weighted cost shown on the right hand side of (31).

Let us choose a control action $\mathbf{z}'(t)$ satisfying the slackness conditions (20)(21)(22). The corresponding 1-step Lyapunov drift plus a weighted cost achieved satisfies

$$\triangle_1(t) + V\cdot g(t) \leq B + V\cdot g(t) - \delta\big[\sum_{j=1}^{J} Q_j(t) + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}(t)\big]. \tag{32}$$

Since GreFar minimizes the right hand side of (31), the 1-step Lyapunov drift plus a weighted cost achieved by GreFar must be less than or equal to that achieved by $\mathbf{z}'(t)$. In other words, the following inequality can be established

$$\triangle_1^*(t) \leq B + V\cdot(g^{\max} - g^{\min}) - \delta\big[\sum_{j=1}^{J} Q_j(t) + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}(t)\big], \tag{33}$$

where $g^{\max}$ and $g^{\min}$ are the maximum and minimum 1-step costs,[11] respectively, and $\triangle_1^*(t)$ is the 1-step Lyapunov drift achieved by GreFar. Now, we define

$$P \triangleq B + V\cdot(g^{\max} - g^{\min}). \tag{34}$$

Thus, if the sum of the queue lengths, $\sum_{j=1}^{J} Q_j(t) + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}(t)$, is greater than or equal to P/δ, the 1-step Lyapunov drift in (33) is non-positive. Moreover, we can show that the maximum value of the Lyapunov function is $[P/(\sqrt{2}\delta)]^2$ under the constraint that the sum of all the queue lengths is less than or equal to P/δ. Thus, if the Lyapunov function is greater than $[P/(\sqrt{2}\delta)]^2$, the sum of all the queue lengths will be greater than P/δ and the Lyapunov function in the next step will not increase, since the 1-step Lyapunov drift is negative. Nevertheless, if the sum of all the queue lengths is less than or equal to P/δ during time t, we have

$$\begin{aligned}L(\Theta(t+1)) \leq & \frac{1}{2}\sum_{j=1}^{J}\left[Q_j(t) + Q_j^{diff}(t)\right]^2 \\ & + \frac{1}{2}\sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j}\left[q_{i,j}(t) + q_{i,j}^{diff}(t)\right]^2 \\ \leq & L(\Theta(t)) + D + \frac{q^{\max}P}{\delta} \\ \leq & \big(\frac{P}{\sqrt{2}\delta}\big)^2 + D + \frac{q^{\max}P}{\delta},\end{aligned} \tag{35}$$

where $Q_j^{diff}(t)$ and $q_{i,j}^{diff}(t)$ represent the absolute values of changes in the respective queue lengths, with maximum values being $Q_j^{diff} = \max[a_j^{max}, \sum_{i\in\mathcal{D}_j} r_{i,j}^{\max}]$ and $q_{i,j}^{diff} = \max[r_{i,j}^{\max}, h_{i,j}^{\max}]$, respectively, $q^{\max} =$

[11] Both $g^{\max}$ and $g^{\min}$ are finite due to boundedness conditions (4)(5).

$\max_{i\in\mathcal{D}_j,j=1,2,\cdots,J}\{Q_j^{diff}, q_{i,j}^{diff}\}$, and D is a constant satisfying, for all $t=0,1,\cdots,t_{end}-1$,

$$\begin{aligned} D \geq & \frac{1}{2}\sum_{j=1}^{J} Q_j^{diff} \max\left[a_j(t), \sum_{i\in\mathcal{D}_j} r_{i,j}(t)\right] \\ & + \frac{1}{2}\sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}^{diff} \max\left[r_{i,j}(t), h_{i,j}(t)\right], \end{aligned} \tag{36}$$

which is finite due to the boundedness conditions. Clearly, $L(\Theta(0))$ satisfies (35), as all the queue lengths are initially zero. Then, by mathematical induction, we can show that, for any $t=0,1,\cdots,t_{end}-1$,

$$L(\Theta(t)) \leq \left(\frac{P}{\sqrt{2}\delta}\right)^2 + D + \frac{q^{\max}P}{\delta}, \tag{37}$$

following which we see that all the queue lengths are bounded by

$$\begin{aligned} Q_j(t), q_{i,j}(t) \leq & \sqrt{\left(\frac{P}{\delta}\right)^2 + 2D + \frac{2q^{\max}P}{\delta}} \\ = & \frac{V\sqrt{\frac{P^2}{V^2} + \frac{2D\delta^2}{V^2} + \frac{2q^{\max}\delta P}{V^2}}}{\delta} = \frac{VC_3}{\delta}, \end{aligned} \tag{38}$$

where

$$C_3 = \sqrt{D_1 + D_2 + D_3}, \tag{39}$$

in which

$$D_1 \triangleq \left[\frac{B}{V} + g^{\max} - g^{\min}\right]^2, \tag{40}$$

$$D_2 \triangleq \frac{2D\delta^2}{V^2}, \tag{41}$$

$$D_3 \triangleq \frac{2q^{\max}\delta}{V}\sqrt{D_1}. \tag{42}$$

This proves part (a) of Theorem 1.

Part (b): Based on (31), we can show that, for $r = 0,1,\cdots,R-1$, the T-step drift plus weighted cost satisfies

$$\begin{aligned} & \triangle_T^*(rT) + V\sum_{t=rT}^{rT+T-1} g^*(t) \\ \leq & BT + V\sum_{t=rT}^{rT+T-1} g(t) \\ & + \sum_{j=1}^{J}\sum_{t=rT}^{rT+T-1}(t-rT)Q_j^{diff}[a_j(t) - \sum_{i\in\mathcal{D}_j} r_{i,j}(t)] \\ & + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j}\sum_{t=rT}^{rT+T-1}(t-rT)q_{i,j}^{diff}[r_{i,j}(t) - h_{i,j}(t)] \\ & + \sum_{j=1}^{J} Q_j(rT)\sum_{t=rT}^{rT+T-1}[a_j(t) - \sum_{i\in\mathcal{D}_j} r_{i,j}(t)] \\ & + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}(rT)\sum_{t=rT}^{rT+T-1}[r_{i,j}(t) - h_{i,j}(t)]. \end{aligned} \tag{43}$$

Then, after some simple mathematic manipulations based on (43), we can derive the following inequality

$$\begin{aligned} & \triangle_T^*(rT) + V\sum_{t=rT}^{rT+T-1} g^*(t) \\ \leq & BT + V\sum_{t=rT}^{rT+T-1} g(t) + DT(T-1) \\ & + \sum_{j=1}^{J} Q_j(rT)\sum_{t=rT}^{rT+T-1}[a_j(t) - \sum_{i\in\mathcal{D}_j} r_{i,j}(t)] \\ & + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}(rT)\sum_{t=rT}^{rT+T-1}[r_{i,j}(t) - h_{i,j}(t)], \end{aligned} \tag{44}$$

where D is a finite constant satisfying (36). In (44), the left hand side is the T-step Lyapunov drift plus weighted cost achieved by GreFar, which explicitly minimizes the right hand side of (31). Note that the right hand side of (31) is smaller than or equal to that of (44). Thus, by considering the optimal T-step lookahead policy on the right hand side of (44), we obtain the following inequality

$$\begin{aligned} & \triangle_T^*(rT) + V\sum_{t=rT}^{rT+T-1} g^*(t) \\ \leq & BT + VTG_r^* + DT(T-1) \\ & + \sum_{j=1}^{J} Q_j(rT)\sum_{t=rT}^{rT+T-1}[a_j(t) - \sum_{i\in\mathcal{D}_j} r_{i,j}(t)] \\ & + \sum_{j=1}^{J}\sum_{i\in\mathcal{D}_j} q_{i,j}(rT)\sum_{t=rT}^{rT+T-1}[r_{i,j}(t) - h_{i,j}(t)] \\ \leq & BT + VTG_r^* + DT(T-1), \end{aligned} \tag{45}$$

where the second inequality follows from the constraints in (16) and (17) satisfied by the optimal T-step lookahead policy. Therefore, by summing (45) over $r=0,1,\cdots,R-1$, and considering that all the queues are initially empty, it follows that

$$\begin{aligned} V\sum_{t=0}^{RT-1} g^*(t) \leq & BTR + VT\sum_{r=0}^{R-1} G_r^* \\ & + RDT(T-1) - L(\Theta(RT-1)) \\ \leq & BTR + VT\sum_{r=0}^{R-1} G_r^* + RDT(T-1). \end{aligned} \tag{46}$$

Finally, by dividing both sides in (46) by VTR, we have

$$\bar{g}^* = \frac{1}{TR}\sum_{t=0}^{RT-1} g^*(t) \leq \frac{1}{R}\sum_{r=0}^{R-1} G_r^* + \frac{B + D(T-1)}{V}, \tag{47}$$

which shows that the online algorithm can achieve a cost within $O(1/V)$ to the minimum cost achieved by the optimal T-step lookahead policy. This proves part (b) of Theorem 1. ■