2014 IEEE 34th International Conference on Distributed Computing Systems

Energy-Efficient Flow Scheduling and Routing with Hard Deadlines in Data Center Networks

Lin Wang[1,2,6], Fa Zhang[1,2], Kai Zheng[3], Athanasios V. Vasilakos[4], Shaolei Ren[5], and Zhiyong Liu[2,7]
[1]Key Laboratory of Intelligent Information Processing, Chinese Academy of Sciences
[2]Institute of Computing Technology, Chinese Academy of Sciences
[3]IBM China Research Lab [4]University of Western Macedonia, Greece
[5]Florida International University, USA [6]University of Chinese Academy of Sciences
[7]State Key Laboratory for Computer Architecture, Institute of Computing Technology, CAS

***Abstract*—The power consumption of enormous network devices in data centers has emerged as a big concern to data center operators. Despite many traffic-engineering-based solutions, very little attention has been paid on performance-guaranteed energy saving schemes. In this paper, we propose a novel energy-saving model for data center networks by scheduling and routing "deadline-constrained flows" where the transmission of every flow has to be accomplished before a rigorous deadline, being the most critical requirement in production data center networks. Based on speed scaling and power-down energy saving strategies for network devices, we aim to explore the most energy efficient way of scheduling and routing flows on the network, as well as determining the transmission speed for every flow. We consider two general versions of the problem. For the version of only flow scheduling where routes of flows are pre-given, we show that it can be solved polynomially and we develop an optimal combinatorial algorithm for it. For the version of joint flow scheduling and routing, we prove that it is strongly NP-hard and cannot have a Fully Polynomial-Time Approximation Scheme (FPTAS) unless P=NP. Based on a relaxation and randomized rounding technique, we provide an efficient approximation algorithm which can guarantee a provable performance ratio with respect to a polynomial of the total number of flows.**

I. Introduction

Cloud computing has become a fundamental service model for the industry. In order to provide sufficient computing resources in clouds, large-scale data centers have been extensively deployed by many companies such as Google, Amazon and Microsoft. While providing powerful computing ability, those data centers are bringing a significant level of energy waste due to the inefficient use of hardware resources, resulting in both high expenditure and environmental concern.

Obviously, the servers should be the first target for energy reduction as they are the most energy-consuming component in a data center. By involving techniques such as Dynamic Voltage Frequency Scaling (DVFS) or hardware virtualization, the energy efficiency of servers has been improved to a large extent. As a result, the network device, as the second-place energy consumer, has taken a large portion in the total energy expenditure of a data center, bringing about urgent economic concerns over data center operators.

The problem of improving the network energy efficiency in data centers has been extensively explored (e.g., [1], [2], [3], [4], [5]). Despite some energy-efficient network topologies for data centers, most of the solutions are concentrated on traffic engineering which aims to consolidate network flows and turn off unused network devices. The essential principle underlying this approach is that data center networks are usually designed with a high level of connectivity redundancy to handle traffic peak and that the traffic load in a data center network varies significantly over time. Due to the fact that the idle power consumed by the chassis usually takes more than half of a switch's total power consumption [6], turning off the switch during idle period should give the most power reduction in theory.

However, the practicality of these aforementioned solutions are quite limited because of the following two aspects: i) Most of the traffic-engineering-based approaches are ineluctably dependent on traffic prediction which seems not feasible or not precise enough [7]. This is because the traffic pattern in a data center network largely depends on the applications running in the data center. Without precise traffic prediction, the network configuration generated by the energy-saving unit has to be updated frequently. Consequently, the network will be suffering from oscillation; ii) Saving energy leads to performance degradation. Most of the solutions only focus on energy efficiency without considering the network performance (e.g. throughput, delay). This will dramatically bring down the reliability of the network, which is not acceptable in practice as providing high performance is the primary goal in a network.

In order to overcome the above two limitations, we propose to view the network traffic from the application-level aspect instead of making use of the static network status (loads on links) that is rapidly monitored from the network or predicted (e.g. [1]). We observe that while the aggregate traffic load varies over time, the most critical factor that conditions the performance of many data exchanges in data centers is meeting flow deadlines ([8], [9], [10], [11], [12]). This is due to the fact that representative data center applications such as search and social networking usually generate a large number of small requests and responses across the data center that are combined together to perform a user-requested computation. As user-percieved performance is evaluated by the speed at which the responses to all the requests are collected and delivered to users, short or guaranteed latency for each of the short request/response flow is strongly required. Given a threshold for tolerable response latency, the system efficiency will be

1063-6927/14 $31.00 © 2014 IEEE
DOI 10.1109/ICDCS.2014.33

definitely conditioned by the number of flows whose deadlines are met (that are completed within the time threshold).

Inspired by this observation, we consider to represent the networking requirements of applications as a set of deadline-constrained flows[1] and we aim to design particular energy-efficient scheduling and routing schemes for them. Although the job scheduling on single or parallel processors with deadline constrains has been extensively studied, little attention has been paid on the job scheduling problem on a multi-hop network [13], especially with the objective of optimizing the energy consumption. To the best of our knowledge, this is the first solution that theoretically explores energy-efficient schemes by scheduling and routing deadline-constrained flows in data center networks.

To summarize our main contributions in this paper: $i)$ We describe the deadline-constrained network energy saving problem and provide comprehensive models for two general versions of this problem – Deadline-Constrained Flow Scheduling (DCFS) and Deadline-Constrained Flow Scheduling and Routing (DCFSR); $ii)$ We show that DCFS can be optimally solved in polynomial time and we propose an optimal combinatorial algorithm for it; $iii)$ We show by in-depth analysis that solving DCFSR is strongly NP-hard and cannot have a fully Polynomial-Time Approximation Scheme (FPTAS) unless P=NP; $iv)$ We provide an efficient approximation algorithm which solves the problem with a provable performance ratio.

The remainder of this paper is organized as follows. Section II presents the modeling for the deadline-constrained network energy saving problem where two versions of the problem are introduced. Section III discusses the DCFS problem where an optimal combinatorial algorithm is provided to solve it. Section IV discusses the DCFSR problem and presents some complexity and hardness analysis. Section V presents an approximation algorithm with guaranteed performance ratio for DCFSR where some numerical results are also provided. Section VI summarizes related work and section VII concludes the paper.

II. The Model

Based on some preliminary definition, we provide the general modeling for the deadline-constrained network energy saving problem in this section.

A. The Data Center

We model a data center as a distributed computing system where a set of servers is connected with a network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of nodes (switches and hosts) and $\mathcal{E}$ is the set of network links. We assume all the switches, as well as all the links, in $\mathcal{V}$ are identical which is reasonable because advanced data center networks such as fat-tree [14] or BCube [15] are usually conducted on identical commodity switches. We use the classical queueing model for links, that is, a link is modeled as a forwarding unit with buffers at its two ends. When the switch finishes processing a data packet, the egress port for this packet will be determined and this packet will be injected into the buffer of the egress link. The packets that queue in the buffer will be transmitted in order according to some preset packet scheduling policy.

We consider the power consumption of network components such as ports and links which are the main power consumers that can be manipulated for energy conservation.[2] With a slight abuse of notation, the power consumption of the ports at the ends of a link is also abstracted into the power consumption of the link for the ease of exposition. For the power consumption model, we adopt the power function from [16] which is an integration of the power-down and the speed scaling model that has been widely used in the literature. For each link $e \in \mathcal{E}$, a power consumption function $f_e(x_e)$ is given to characterize the manner in which energy is being consumed with respect to the transmission rate x_e of link e. We assume uniform power functions as the network is composed of identical switches and links. Formally, for every link we are given a function $f(\cdot)$ which is expressed by

$$f(x_e) = \begin{cases} 0 & x_e = 0 \\ \sigma + \mu x_e^{\alpha} & 0 < x_e \leq C \end{cases}, \tag{1}$$

where σ, μ and α are constants associated with the link type. Constant σ is known as the idle power for maintaining link states, while C is the maximum transmission rate of a link. Normally, the power function $f(\cdot)$ is superadditive, i.e., $\alpha > 1$. In order to get rid of the network stability problem introduced by frequently togging on and off links, we assume that a link can be turned off only when it carries no traffic during the whole given period of time. Making this assumption also helps reduce the considerable power incurred by changing the state of a link, as well as the wear-and-tear cost.

B. Applications

We model an application as a set of deadline-constrained flows each of which consists of a certain amount of data that has to be routed from a source to a destination on the network within a given period of time. In order to avoid packet reordering at the destination end, we assume that each flow can only follow a single path. Nevertheless, multi-path routing protocols can be incorporated in our model by splitting a big flow into many small flows with the same release time and deadline at the source end and each of the small flows will follow a single path.

Let $[T_0, T_1]$ be a fixed time interval, during which a set $\mathcal{J} = \{j_1, j_2, \ldots, j_n\}$ of flows has to be routed on the network. Associated with each flow $j_i \in \mathcal{J}$ are the following parameters:

- w_i amount of data that needs to be routed,
- r_i, d_i its release time and deadline respectively, and

[1]If not specified, "flow" in this paper refers to a certain amount of data that has to be transmitted from a source to a destination on the network.

[2]The biggest power consumer in a switch, the chassis, cannot achieve power proportionality easily due to drastic performance degradation. Nevertheless, our approach is a complement and can be incorporated with switch-level power-down based solutions.

- p_i, q_i its source and destination respectively.

Without loss of generality, we assume $T_0 = \min_{j_i \in \mathcal{J}} r_i$ and $T_1 = \min_{j_i \in \mathcal{J}} d_i$. In our setting, we allow preemption, i.e., each flow can be suspended at any time and recovered later. We define $S_i = [r_i, d_i]$ as the *span* of flow j_i and we say that j_i is active at time t if $t \in S_i$. The *density* of flow j_i is defined as $D_i = w_i/(d_i - r_i)$. A schedule is a set

$$\mathcal{S} = \{(s_i(t), \mathcal{P}_i) \mid \forall j_i \in \mathcal{J}, \forall t \in [r_i, d_i]\} \tag{2}$$

where $s_i(t)$ is the transmission rate chosen for flow j_i at time t and $\mathcal{P}_i$ is the set of links that are on the chosen path for carrying the traffic from this flow. A schedule is called feasible if every flow can be accomplished within its deadline following this schedule, i.e., $\mathcal{S}$ satisfies

$$\int_{r_i}^{d_i} s_i(t)dt = w_i, \forall j_i \in \mathcal{J}. \tag{3}$$

We define $\mathcal{E}_a$ as the set of active links where

$$\mathcal{E}_a = \{e \in \mathcal{E} \mid \exists t \in [T_0, T_1], x_e(t) > 0\}. \tag{4}$$

Consequently, the total energy consumed by all the links during $[T_0, T_1]$ in a schedule $\mathcal{S}$ can be expressed by

$$\Phi_f(\mathcal{S}) = (T_1 - T_0) \cdot |\mathcal{E}_a| \cdot \sigma + \int_{T_0}^{T_1} \sum_{e \in \mathcal{E}_a} \mu \left(x_e(t)\right)^\alpha dt \tag{5}$$

where $x_e(t)$ is the transmission rate of link e at time t and $x_e(t) = s_i(t)$ if flow j_i is being transmitted on link e at time t. Our objective is to find a feasible schedule that minimizes $\Phi_f(\mathcal{S})$. Depending on whether the routing protocol is given or not, we have two versions of this problem which we call DCFS (Deadline-Constrained Flow Scheduling) and DCFSR (Deadline-Constrained Flow Scheduling and Routing). We will discuss them separately in the following sections.

III. Deadline-Constrained Flow Scheduling

In this section, we discuss the DCFS problem. Specifically, we model this problem as a convex program and show that it can be optimally solved. We then provide an optimal combinatorial algorithm for it.

A. Preliminaries

In DCFS, the routing paths for all the flows are provided. Routing the flows with these paths, each link will be assigned with a set of flows $\mathcal{J}_e = \{j_i \mid e \in \mathcal{P}_i\}$. We omit those inactive links that satisfy $\mathcal{J}_e = \emptyset$ since they will never be used for transmitting data. Thus the set of active links is $\mathcal{E}_a = \mathcal{E} \setminus \{e \in \mathcal{E} \mid \mathcal{J}_e = \emptyset\}$. As all the links in $\mathcal{E}_a$ will be used, we simplify the problem by replacing the power consumption function with $g(x_e) = \mu x_e^\alpha$. Consequently, the objective of the problem becomes to find a feasible schedule $\mathcal{S}$ such that

$$\Phi_g(\mathcal{S}) = \int_{T_0}^{T_1} \sum_{e \in \mathcal{E}_a} g\left(x_e(t)\right) dt \tag{6}$$

is minimized. For the sake of tractability, we first consider the case where the routing path for each flow is a virtual circuit. That is, when a flow is being routed, all the links on the routing path of this flow will be totally occupied by the packets from this flow. Nevertheless, we will show that this assumption is generally true in the optimal solution and it can be realized by assigning priorities to the packets from each flow in a packet-switching network.

We define the *minimum-energy schedule* as the schedule that minimizes the total power consumption but may not satisfy the maximum transmission rate constraint on each link. Then, we introduce the following lemmas.

Lemma 1. *The minimum-energy schedule will use a single transmission rate for every flow.*

Lemma 2. *The minimum-energy schedule will choose an as small as possible transmission rate for each flow such that the deadlines of flows can be guaranteed.*

Proof: We focus on one flow with an amount w of data that has to be routed in time interval $[r, d]$. The routing path for this flow is denoted by $\mathcal{P}$ and the number of links in $\mathcal{P}$ is given by $|\mathcal{P}|$. Using Lemma 1, we assume a single transmission rate s is given to process this flow. The total energy consumed by the links for routing this flow can be expressed by $\Phi_g = \mu s^\alpha \cdot w/s = \mu \cdot w \cdot s^{\alpha-1}$. As long as $\alpha > 1$, Φ_g is minimized when we have the minimum transmission rate s for this flow such that the deadlines of all the flows can be satisfied. In this sense, the minimum-energy schedule will use the minimized transmission rate for each flow. ■

Following the above lemma, we observe that as long as there are feasible schedules, the minimum-energy schedule is feasible. In other words, the minimum-energy schedule is also the optimal schedule for DCFS. Equivalently, the maximum transmission rate constraint $x_v(t) \le C$ can be relaxed in DCFS. In the remainder of this section, we will omit that constraint.

B. Problem Formulation

We denote the transmission rate for flow j_i as s_i according to Lemma 1. The DCFS problem can be formulated as the following convex program.

$$(P_1) \qquad \min \sum_{e \in \mathcal{E}_a} \sum_{j_i \in \mathcal{J}_v} w_i \cdot \mu s_i^{\alpha-1}$$

subject to

$$\sum_{j_i \in \mathcal{J}'} \frac{w_i}{s_i} - (\max_{j_i \in \mathcal{J}'} d_i - \min_{j_i \in \mathcal{J}'} r_i) \le 0 \qquad \mathcal{J}' \subseteq \mathcal{J}_e$$
$$s_i > 0 \qquad \forall j_i \in \mathcal{J}$$

The total transmission time and the total energy consumption of flow j_i are w_i/s_i and $w_i/s_i \cdot \mu s_i^\alpha = w_i \cdot \mu s_i^{\alpha-1}$, respectively. The first constraint forces that for an arbitrary link e, all the flows in any subset of $\mathcal{J}_e$ has to be processed before the last deadline of the flows in that subset. The second constraint represents that the transmission rate for each flows has to be larger than 0. It is easy to verify that program (P_1) is convex because the objective function is convex (as we assume $\alpha > 1$) while all the constraints are linear.

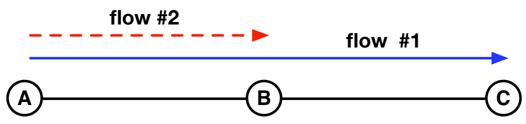

Figure 1. A line network consisting of three nodes connected by two links.

As a result, the DCFS problem can be solved optimally in polynomial time by applying the Ellipsoid Algorithm [17]. However, as the Ellipsoid Algorithm is not practically used due to its high complexity in typical instances, we aim to construct an efficient combinatorial algorithm by exploring the characteristics of the minimum-energy schedule.

C. An Optimal Combinatorial Algorithm

We now provide a combinatorial algorithm which can always find the optimal schedule for DCFS. Before presenting the algorithm, we first give a characterization of the optimal schedule through the following example.

Example 1. *Consider a line network whose topology is given in Fig. 1. The power consumption of the links is characterized by function $f(x_e) = x_e^2$. On this network we have two flows j_1 and j_2 that need to be routed. The details of the two flows are given by the following multi-tuples*

$$j_1 \triangleq (p_1 = A, q_1 = C, r_1 = 2, d_1 = 4, w_1 = 6),$$
$$j_2 \triangleq (p_2 = A, q_2 = B, r_2 = 1, d_2 = 3, w_2 = 8).$$

According to Lemma 1, we denote the transmission rates for j_1 and j_2 as s_1 and s_2, respectively. Consequently, we have the following three constraints $6/s_1 \leq d_1 - r_1 = 2$, $8/s_2 \leq d_2 - r_2 = 2$ and $6/s_1 + 8/s_2 \leq d_1 - r_2 = 3$, while the objective function is $\Phi = 2\times 6\times s_1 + 8\times s_2$. It is easy to check that in the optimal schedule, $\sqrt{2}s_1 = s_2 = \frac{8+6\sqrt{2}}{3}$. We then construct an instance of the speed scaling problem on single processor (SS-SP) raised by Yao *et al.* [18]. Consider we have two jobs with required numbers of CPU cycles of $6\sqrt{2}$ and 8 respectively, while the release times and deadlines are exactly the same as the two flows. Using the **Optimal-Schedule** algorithm (known as the **YDS** algorithm according to the authors' initials), the two jobs will be processed at the same speed of $\frac{8+6\sqrt{2}}{3}$ in time interval $[1, 4]$. As a result, the objective value in the optimal schedule for this instance is exactly the same as the minimum Φ in our problem while the structure of the solution is also the same. Using this observation, we provide an optimal algorithm for solving the DCFS problem based on the **YDS** algorithm.

We first construct from the DCFS problem a variant of the SS-SP problem by introducing a virtual weight $w_i' = w_i \cdot (|\mathcal{P}_i|)^{1/\alpha}$ for each flow $j_i \in J$. First of all, we present some definitions which are extended from [18].

Definition 1. *The intensity of an interval $I = [a, b]$ on a link e is defined by*

$$\delta(I, e) = \frac{\sum_{[r_i,d_i]\subseteq[a,b]\wedge j_i\in\mathcal{J}_e} w_i'}{a \sim b} \tag{7}$$

Algorithm 1 Most-Critical-First

Input: data center network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, set of flows $\mathcal{J}_e$ for $e \in \mathcal{E}$ and virtual weights w_i' for each flow
Output: transmission rate s_i and transmission time interval $[r_i', d_i']$ for each flow $j_i \in \mathcal{J}$

```
1: while $\exists e \in \mathcal{E}, \mathcal{J}_e \neq \emptyset$ do
2:     Find the critical interval $I^*$ and the critical link $e$. The
       flows in this interval can be represented by $\mathcal{J}^* = \{j_i \mid [r_i, d_i] \subseteq I^* \wedge e \in \mathcal{P}_i\}$ and without loss of
       generality,
           $I^* = [a, b] = [\min_{j_i\in\mathcal{J}^*} r_i, \max_{j_i\in\mathcal{J}^*} d_i].$
3:     Schedule the flows in $\mathcal{J}^*$ with the Earliest Deadline
       First (EDF) policy using transmission rate
           $s_i = \frac{\sum_{j_i\in\mathcal{J}^*} w_i'}{(|\mathcal{P}_i|)^{1/\alpha}(a \sim b)}$
       for each flow $j_i \in \mathcal{J}^*$. The transmission time interval
       $[r_i', d_i']$ is also determined.
4:     for $j_i \in \mathcal{J}^*$ do
5:         $\mathcal{J}_e \leftarrow \mathcal{J}_e \setminus j_i$ for $e \in \mathcal{P}_i$.
6:         For $e \in \mathcal{P}_i$, mark the time interval $[r_i', d_i']$ as unavail-
           able on link $e$.
7:     end for
8: end while
```

where $a \sim b$ denotes the available time in interval $[a, b]$.

Intuitively, the following inequality has to be satisfied,

$$\int_a^b x_e(t)dt/(a \sim b) \geq \delta(I, e), \tag{8}$$

which means that $\delta(I, e)$ is a lower bound on the average transmission rate of link e in any feasible schedule over the interval $[a, b]$.

Definition 2. *If an interval $I^* = [a, b]$ maximizes $\delta(I^*, e)$ for any $e \in \mathcal{E}_a$, we call I^* a critical interval and e is the corresponding critical link.*

Now we present the main algorithm that generates optimal schedules greedily by computing critical intervals iteratively. The pseudo-code of the algorithm **Most-Critical-First** is shown in Algorithm 1. The following theorem proves that a critical interval will determine a segment of the optimal schedule.

Theorem 1. *Let I^* be a critical interval and e be the corresponding critical link, algorithm **Most-Critical-First** can guarantee that the energy consumed by routing the flows in $\mathcal{J}^*$ is optimal.*

Proof: Denoting the transmission rate for each flow $j_i \in \mathcal{J}^*$ as s_i, the total energy consumed by routing the flows in $\mathcal{J}^*$ is expressed by

$$\Phi_g(\mathcal{J}^*) = \sum_{j_i\in J^*} |\mathcal{P}_i| \times w_i \times s_i^{\alpha-1}. \tag{9}$$

According to the second constraint in program (P_1) we have

$$\sum_{j_i \in \mathcal{J}^*} \frac{w_i}{s_i} - (a \sim b) \leq 0. \tag{10}$$

It is clear that in the optimal schedule, the above inequality is exactly an equality because of Lemma 2. Then, $\Phi_g(\mathcal{J}^*)$ can be minimized by using the method of Lagrange multipliers. By introducing a Lagrange multiple λ, we construct a function

$$L(s_1, s_2, \ldots, s_{|J^*|}, \lambda) = \Phi_g(\mathcal{J}^*) + \lambda(\sum_{j_i \in \mathcal{J}^*} \frac{w_i}{s_i} - (a \sim b)). \tag{11}$$

By setting $\bigtriangledown L(s_1, s_2, \ldots, s_{|J^*|}, \lambda) = 0$, we have

$$(|\mathcal{P}_1|)^{1/\alpha} s_1 = \ldots = \left(|\mathcal{P}_{|\mathcal{J}^*|}|\right)^{1/\alpha} s_n. \tag{12}$$

This is equivalent to solving an instance of the SS-SP problem as we explained in Example 1, where we treat each flow as a job with weight $w'_i = (|\mathcal{P}_i|)^{1/\alpha} w_i$. Using Theorem 1 provided in [18], we set the processing speed of all the jobs to the same value of $\sum_{j_i \in \mathcal{J}^*} w'_i/(a \sim b)$, which will give the optimal energy consumption for the SS-SP problem, as well as our problem. That is, $\Phi_g(\mathcal{J}^*)$ is minimized by setting

$$(|\mathcal{P}_1|)^{1/\alpha} s_1 = \ldots = \left(|\mathcal{P}_{|\mathcal{J}^*|}|\right)^{1/\alpha} s_n = \frac{\sum_{j_i \in \mathcal{J}^*} w'_i}{(a \sim b)} \tag{13}$$

which is reflected in the algorithm. ∎

Actually, algorithm **Most-Critical-First** solves a variant of the SS-SP problem based on the YDS algorithm. Consequently, the following result follows quickly from Theorem 1.

Corollary 1. *The schedule produced by algorithm **Most-Critical-First** is optimal to the DCFS problem.*

The time complexity of algorithm **Most-Critical-First** is bounded by $O(n^2|\mathcal{V}|)$. Note that the optimality of this algorithm is maintained based on the assumption that data in flows is routed exclusively through virtual circuits. We now show how to extend it to a packet-switching network: we assign a unique priority for all the packets from each flow according to the flow's starting time r'_i. That is, a flow j_i with a smaller r'_i will have a higher priority. This priority information can be encapsulated into the header of each packet and links will schedule those packets according to their priorities.

IV. Deadline-Constrained Flow Scheduling and Routing

In this section, we discuss the DCFSR problem. We aim at exploring the most energy-efficient scheduling and routing scheme for a given collection of flows. This problem is much harder than DCFS as we have to decide also the routing path for each flow, as well as the transmission rate.

A. Problem Formulation

We observe that once we have the routing paths for all flows determined, finding the transmission rate for each flow is then the DCFS problem which can be optimally solved by algorithm **Most-Critical-First**. Let $\mathcal{P}_i$ denote the routing path for flow j_i. Keeping the notation we used before, the DCFSR problem can be formalized by the following program.

$$(P_2) \qquad \min \Phi_f$$

subject to

$$\begin{aligned} &\int_{r_i}^{d_i} s_i(t)dt \geq w_i && \forall j_i \in \mathcal{J} \\ &s_i(t) \leq x_e(t) && \forall e \in \mathcal{P}_i \\ &0 < x_e(t) \leq C && \forall e \in \mathcal{E} \\ &s_i(t) : \text{flow conservation} \end{aligned}$$

The first constraint represents that each flow has to be finished before its deadline. The second constraint means that the transmission rate of the flow that is being processed on a link e cannot exceed the operation rate of that link, while the third one represents that the operation rate of a link has to be larger than zero and no larger than the maximum operation rate C. The flow conservation in the last constraint forces that $\mathcal{P}_i$ is a path connecting source p_i and destination q_i of flow j_i.

B. Complexity and Hardness Results

First, we provide the following definition and lemma as preliminaries based on our power consumption model.

Definition 3. *The **power rate** of a link e ($x_e > 0$) is defined as the power consumed by each unit of traffic, i.e., $f(x_e)/x_e$.*

It can be observed that as long as the power rate of every link is minimized, the total power consumption of the network will be optimal. To minimize the power rate of a link, we show the following lemma.

Lemma 3. *Ideally, the optimal operation rate R_{opt} for a link is given by $R_{opt} = \left(\frac{\sigma}{\mu(\alpha-1)}\right)^{1/\alpha}$.*

Note that this operation rate is optimal in theory but is not always achievable in practice, as it can happen that $R_{opt} > C$. In general, we can prove that the decision version of DCFSR is NP-complete by providing the following theorem.

Theorem 2. *Given a certain amount of energy Φ_0, finding a schedule $\mathcal{S}$ for DCFSR such that $\Phi(\mathcal{S}) \leq \Phi_0$ is NP-complete.*

Proof: This can be proved by a simple reduction from the 3-partition problem which is NP-complete [19]. Suppose we are given an instance of the 3-partition problem with a set $\mathcal{A}$ of $3m$ integers $a_1, a_2, \ldots, a_{3m}$ where $\sum_{i=1}^{3m} a_i = mB$ and $a_i \in [B/4, B/2]$. The problem is to decide whether $\mathcal{A}$ can be partitioned into m disjoint subsets, i.e., $\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2 \cup \ldots \cup \mathcal{A}_m$ and $\mathcal{A}_i \cap \mathcal{A}_j = \emptyset$ for any $i, j \in m$, such that every subset $\mathcal{A}_i$ consists of 3 integers and $\sum_{a \in \mathcal{A}_i} a = B$. Based on this 3-partition instance, we construct an instance of DCFSR as follows: we are given a network where two nodes (denoted as *src* and *dst*) are connected in parallel by k ($k >> m$) links. Assume we are given a set $\mathcal{J}$ of $3m$ flows each of which has an amount a_i ($i \in [1, 3m]$) of data needed to be transmitted from *src* to *dst* on the network. All the flows arrive at the same time and the data transmission has to be finished in one unit of time. We assume $B < C$ and $\sigma = \mu(\alpha-1)B^{\alpha}$, i.e., $R_{opt} = B$ and we set $\Phi_0 = m \cdot \alpha\mu B^{\alpha}$. We will show that there is a schedule $\mathcal{S}$ such that $\Phi(\mathcal{S}) \leq \Phi_0$ if and only if $\mathcal{A}$ can be partitioned in the way as in the optimal solution of the 3-partition instance.

On the one hand, if there exists a partition for the 3-partition instance, we have a solution $\mathcal{S}$ for the DCFSR instance where the flows are transmitted by m links each with an operation rate B according to the partition and the energy consumption in this solution is $\Phi(\mathcal{S}) = m \cdot \alpha\mu B^{\alpha}$. According to Lemma 3, this solution is optimal since the power rate for each link in this solution is optimal. Hence, it satisfies that $\Phi(\mathcal{S}) \leq \Phi_0$. On the other hand, if we obtain a solution $\mathcal{S}$ for the DCFSR instance such that $\Phi(\mathcal{S}) \leq \Phi_0$. It can then be derived that exactly m links will be used and each link will use an operation rate B. Otherwise, the total energy consumption $\Phi(\mathcal{S})$ will be larger than Φ_0 as the average power rate of the used links is larger than $f(B)/B$, so $\Phi(\mathcal{S}) > f(B)/B \cdot mB = m \cdot \alpha\mu B^{\alpha}$. Accordingly, we can construct a partition for the 3-partition instance. In a nutshell, finding a partition for 3-partition is equivalent to finding a solution $\mathcal{S}$ for DCFSR such that $\Phi(\mathcal{S}) \leq \Phi_0$.

The above reduction is based on the assumption that $R_{opt} < C$, which is not necessarily true in reality. However, in the case $R_{opt} > C$, we just set $B = C$ and $\Phi_0 = m(\sigma + \mu C^{\alpha})$, and the same reduction can be built in a similar way. ■

Then, it follows directly that

Corollary 2. *Solving the DCFSR problem is strongly NP-hard.*

As a result, the DCFSR problem can only be solved by approximating the optimum. When we say that an algorithm approximates DCFSR with performance ratio γ, it means that the energy consumption in the solution produced by this algorithm is at most γ times the minimum energy consumption. Given these, we aim at designing an algorithm to approximate the optimum with ratio γ as small as possible. Unfortunately, for the case $R_{opt} > C$, which is likely to be the real situation as justified in [5], the following theorem shows that γ cannot be as small as we want since there is a lower bound for it.

Theorem 3. *There exists an instance of problem DCFSR such that no approximation algorithm can guarantee a performance ratio smaller than* $\frac{3}{2}\left(1 + \frac{(2/3)^{\alpha}-1}{\alpha}\right)$ *unless P=NP.*

Proof: We prove this theorem by showing a gap-preserving reduction from the partition problem which is NP-complete. Suppose we are given an instance of the partition problem with a set $\mathcal{A}$ of n integers. Assuming $\sum_{a\in\mathcal{A}} a = B$, the problem is whether it is possible to find a subset $\mathcal{A}' \subset \mathcal{A}$ such that $\sum_{a\in\mathcal{A}'} a = B/2$. We now construct an instance of the DCFSR problem as follows: we consider also the same network as the one in the previous proof where we assume $m > 2$ and the capacity of each link is given by $C = B/2$. We are also given a set $\mathcal{J}$ of flows each of which requires to route an amount w_i of data from *src* to *dst* and $w_i = \mathcal{A}[i]$ for $j_i \in \mathcal{J}$. These flows arrive at the system at the same moment and have to be accomplished in one unit time. We denote by Φ_{opt} the optimal solution of the DCFSR instance, which represents the total energy consumed by the active links for tranmisstting the flows. Then, the following properties are preserved.

$$\exists\mathcal{A}' \subset \mathcal{A}, \sum_{a\in\mathcal{A}'} a = B/2 \Longrightarrow \Phi_{opt} = 2b + 2\mu C^{\alpha},$$
$$\nexists\mathcal{A}' \subset \mathcal{A}, \sum_{a\in\mathcal{A}'} a = B/2 \Longrightarrow \Phi_{opt} \geq 3b + 3\mu(2C/3)^{\alpha}.$$

Comparing both optimal solutions, we obtain a ratio γ where

$$\gamma = \frac{3\sigma + 3\mu\,(2C/3)^{\alpha}}{2\sigma + 2C^{\alpha}} \geq \frac{3\mu C^{\alpha}(\alpha-1) + 3\mu\,(2C/3)^{\alpha}}{2\mu C^{\alpha}(\alpha-1) + 2\mu C^{\alpha}}$$
$$= \frac{3}{2}\left(1 + \frac{(2/3)^{\alpha}-1}{\alpha}\right) \quad (14)$$

where the inequality is obtained by applying $\sigma \geq \mu C^{\alpha}(\alpha-1)$. Combining with the two properties we derived, it is easy to conclude that as long as there is an approximation algorithm solving DCFSR with a performance ratio better than γ, a subset $\mathcal{A}'$ in the partition problem can be found such that $\sum_{a\in\mathcal{A}'} a = B/2$. However, it is well known that the partition problem is NP-complete and cannot be solved by any polynomial time algorithm optimally. As a result, no algorithm can approximate DCFSR with a performance ratio better than γ unless P=NP. ■

This directly implies that the DCFSR cannot have Fully Polynomial-Time Approximation Schemes (FPTAS) under the conventional assumption that P≠NP [20]. In the next section, we will provide an efficient approximation algorithm.

V. An Approximation Algorithm for DCFSR

We present an approximation algorithm for DCFSR in this section. This algorithm is based on a relaxation and randomized rounding based process. We show by both analysis and numerical results that this approximation algorithm can guarantee a provable performance ratio.

A. The Algorithm

We first provide the following preliminaries. We define $\mathcal{T} = \{t_0, t_1, \ldots, t_K\}$ to be the set of release times and deadlines of all the flows such that $t_{k_1} < t_{k_2}$ for any $0 \leq k_1 < k_2 \leq K$. It is clear that $t_0 = \min_{j_i\in\mathcal{J}}\{r_i\}$ and $t_K = \min_{j_i\in\mathcal{J}}\{d_i\}$. We denote by I_k the time interval $[t_{k-1}, t_k]$ for $1 \leq k \leq K$, by $|I_k|$ the length of interval I_k and by $\beta_k = |I_k|/(t_K - t_0)$ the fraction of time that interval I_j takes from the whole time of interest. We also define $\lambda = (t_K - t_0)/\min_k |I_k|$.

We first relax the problem by making the following transformations such that the resulted problem can be optimally solved.

- The traffic load of flow j_i is given by its density D_i. Flows can be routed simultaneously on any link;
- Each flow can be routed through multiple paths;
- The links in the network can be flexibly turned on and off at any moment.

We observe that the resulted problem can be decomposed into a set of subproblems in each interval I_k as in each interval I_k, the traffic flows on the network are invariable. Actually, each such subproblem in an interval is a fractional multi-commodity flow (F-MCF) problem that is precisely defined as follows.

Definition 4 (F-MCF). *For every active flow $j_i \in \mathcal{J}$ that satisfies $I_k \subseteq S_i$, a flow of traffic load D_i has to be routed from p_i to q_i. The objective is to route these flows on the network such that the total cost on the links is minimized, given cost function $f(\cdot)$ for every link.*

It is known that the F-MCF problem can be optimally solved by convex programming. Consequently, we obtain the fractional solution $y^*_{i,e}(k)$ which represents the proportion of the amount of the flow j_i that goes through link e in interval I_k. Absolutely, this solution is not feasible to the original DCFSR problem. Now we aim to transform this infeasible solution into a feasible one.

The transformation is accomplished by a randomized rounding process. Before that, we extract candidate routing paths for each flow following the Raghavan-Tompson [21] manner as follows. For each interval I_k ($1 \le k \le K$), we decompose the fractional solution $y^*_{i,e}(k)$ into weighted flow paths for each flow j_i via the following procedure. We repeatedly extract paths connecting the source and destination of each flow j_i from the subgraph defined by links e for which $y^*_{i,e}(k) > 0$. For each extracted path $\mathcal{P}$, we assign a weight $w_{\mathcal{P}} = \min_{e \in \mathcal{P}} y^*_{i,e}(k)$ for $\mathcal{P}$ and the value of $y^*_{i,e}(k)$ on every link in $\mathcal{P}$ is reduced by $w_{\mathcal{P}}$. This path extracting process will terminate when every $y^*_{i,e}(k)$ becomes zero, which is guaranteed by the flow conservation constraint. As a result, we obtain a set $\mathcal{Q}_i(k)$ of paths for flow j_i in interval I_k. We repeat this process for every interval and denote $\mathcal{Q}_i = \cup_{1 \le k \le K} \mathcal{Q}_i(k)$ as the set of all the candidate paths for flow j_i without duplication. Note that a path $\mathcal{P}$ may be used in more than one interval. We denote by $w_{\mathcal{P}}(k)$ the corresponding weights of $\mathcal{P}$ in different intervals. If $\mathcal{P}$ is not used in interval I_k, then $w_{\mathcal{P}}(k) = 0$.

Now we show how to choose a single path for each flow j_i from the candidate paths $\mathcal{Q}_i$. For each path $\mathcal{P} \in \mathcal{Q}_i$, we assign a new weight $\bar{w}_{\mathcal{P}}$ where $\bar{w}_{\mathcal{P}} = \sum_k w_{\mathcal{P}}(k) \cdot |I_k|/(d_i - r_i)$. The routing path $\mathcal{P}_i$ for flow j_i is then determined by randomly choosing a path $\mathcal{P}$ from $\mathcal{Q}_i$ using weight $\bar{w}_{\mathcal{P}}$ as the probability at which path $\mathcal{P}$ will be chosen. This path choosing process will be repeated for every flow. Consequently, a single path $\mathcal{P}_i$ will be determined for each flow $j_i \in \mathcal{J}$ and the packets from this flow will be routed through only this path.

Finally, we choose a transmission rate for each flow in every interval I_k. Denoting also $\mathcal{J}_e(k)$ the flow that will be transmitted on link e in interval I_k, the transmission rate for every flow $j_i \in \mathcal{J}_e(k)$ will be set to $\sum_{j_i \in \mathcal{J}_e(k)} D_i$ and data packets from each flow in $\mathcal{J}_e(k)$ will be forwarded on e using the EDF policy which we have introduced before.

The whole process of the algorithm is shown in Algorithm 2. We have to mention that the proposed algorithm does not guarantee the maximum operation rate constraint. However, we observe that the probability that many flows are simultaneously requested to be forwarded on a designated link in the proposed algorithm is very low as the probability for choosing a link for a flow is derived from the fractional solution which has the maximum operation rate constraint considered. Nevertheless, we can always repeat the randomized rounding process until we obtain a feasible solution. Now we show that

Algorithm 2 Random-Schedule

```
Input: data center network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, set of flows $\mathcal{J}$
Output: Routing path $\mathcal{P}_i$ for flows $j_i \in \mathcal{J}$ and transmission rate $s_i(t)$ for $t \in [r_i, d_i]$
 1: Transform the DCFSR problem into a multi-step fractional multi-commodity flow problem by relaxing the constraints
 2: for $I_k \in \{I_1, \ldots, I_K\}$ do
 3:     Solve the fractional multi-commodity flow problems by convex programming, obtaining $y^*_{i,e}(k)$
 4:     Extract candidate paths for each flow, denote as $\mathcal{Q}_i(k)$ and a weight $w_{\mathcal{P}}(k)$ for each $\mathcal{P} \in \mathcal{Q}_i(k)$
 5: end for
 6: $\mathcal{Q}_i = \cup_{1 \le k \le K} \mathcal{Q}_i(k)$ for $j_i \in \mathcal{J}$
 7: $\bar{w}_{\mathcal{P}} = \sum_k w_{\mathcal{P}}(k) \cdot |I_k|/(d_i - r_i)$ for $\mathcal{P} \in \mathcal{Q}_i$
 8: for $j_i \in \mathcal{J}$ do
 9:     Randomly choose a path $\mathcal{P}_i$ from $\mathcal{Q}_i$ using weight $\bar{w}_{\mathcal{P}}$ as the probability
10: end for
11: Route the packets from all the flows on link $e$ in interval $I_k$ using the EDF policy. The transmission rate for flow $j_i$ on link $e$ in interval $I_k$ is $\sum_{j_i \in \mathcal{J}_e(k)} D_i$.
```

Theorem 4. *The deadline of every flow $j_i \in \mathcal{J}$ can be met in the solution produced by algorithm **Random-Schedule**.*

Proof: As we allow preemption for each flow, it suffices to show that all the data that arrives on a link e in every interval $I_k \subseteq S_j$ can be transmitted by the end of this interval. Let us focus on one arbitrary link e and an arbitrary interval I_k. The total amount of data that has to be transmitted through e in this interval is given by $\sum_{j_i \in \mathcal{J}_e(k)} (D_i \cdot |I_k|)$. As the transmission rate for every flow in $\mathcal{J}_e(k)$ is $\sum_{j_i \in \mathcal{J}_e(k)} D_i$, the total time that is needed for accomplishing all the flows is equal to $\sum_{j_i \in \mathcal{J}_e(k)} (D_i \cdot |I_k|) \,/ \sum_{j_i \in \mathcal{J}_e(k)} D_i = |I_k|$. As a result, all the data in this interval can always be transmitted no matter what kind of scheduling policy is used. However, we use the EDF policy because it can significantly reduce the frequency of changing the transmission rates of links. ∎

B. Performance Analysis

We now analyze the approximation performance of the proposed algorithm. Our results are based on the main results in [16] where the authors also used a rounding process to approximate the multi-commodity flow problem with $f(\cdot)$. The biggest difference compared with that work is that the rounding process we propose in this paper is responsible for minimizing the number of used links and thus has to guarantee the same path for each flow in every interval. That is, we aim at solving a multi-step MCF problem. We base our proof on the following result and we only show the difference from it.

Theorem 5 ([16])**.** *For nonuniform demands, randomized rounding can be used to achieve a $O\left(K + \log^{\alpha-1} D\right)$ – approximation for MCF with cost function $f(\cdot)$, where K is the total number of demands and D is the maximum demand among all the demands.*

Theorem 6. *Algorithm* ***Random-Schedule*** *can achieve a $O\left(\lambda^\alpha (n^2 \log D)^{\alpha-1}\right)$-approximation for the DCFSR problem with power function $g(x) = \mu x^\alpha$, where $D = \max_{j_i \in \mathcal{J}} D_i$.*

Proof: We follow the process of the proof for Theorem 5. First, we assume unit flows, i.e., $w_i/(d_i - r_i) = 1$ and we use power consumption function $h(x) = \max\{\mu x, \mu x^\alpha\}$. Note here that we have $E(\hat{x}_e(l)) \leq \lambda \sum_k \beta_k x_e^*(k)$ for any $1 \leq l \leq K$. We then consider the following two cases:
Case 1: $\sum_{1\leq k\leq K} \beta_k x_e^*(k) \leq 1$, we have,

$$\begin{aligned} & E(\sum_{1\leq l\leq K} g(\hat{x}_e(l))|I_l|) \leq \sum_{1\leq l\leq K} \gamma_1 h(E(\hat{x}_e(l)))|I_l| \\ \leq\ & \gamma_1 \lambda (t_K - t_0) \sum_{1\leq k\leq K} \beta_k \mu x_e^*(k) \\ =\ & \gamma_1 \lambda \sum_{1\leq k\leq K} h(x_e^*(k))|I_k| \qquad (15) \\ =\ & \gamma_1 \lambda \sum_{1\leq k\leq K} \sum_{j_i\in J} h(x_{i,e}^*(k))|I_k|, \end{aligned}$$

where the first inequality follows from the result in [16].
Case 2: $\sum_{1\leq k\leq K} \beta_k x_e^*(k) > 1$, we have,

$$\begin{aligned} & E(\sum_{1\leq l\leq K} g(\hat{x}_e(l))|I_l|) \leq \sum_{1\leq l\leq K} \gamma_2 h(E(\hat{x}_e(l)))|I_l| \\ \leq\ & \gamma_2 \lambda^\alpha K^{\alpha-1} (t_K - t_0) \sum_{1\leq k\leq K} \beta_k^\alpha \mu (x_e^*(k))^\alpha \\ \leq\ & \gamma_2 \lambda^\alpha K \sum_{1\leq k\leq K} h(x_e^*(k))|I_k| \qquad (16) \\ \leq\ & 2\gamma_2 \lambda^\alpha n^{2(\alpha-1)} \sum_{1\leq k\leq K} \sum_{j_i\in J} h(x_{i,e}^*(k))|I_k|, \end{aligned}$$

where the first inequality follows also from the result in [16]. The second inequality and the last inequality follow from the property that $(x_1 + \ldots + x_m)^\alpha \leq m^{\alpha-1}(x_1^\alpha + \ldots + x_m^\alpha)$ while in the last inequality, we also apply $K \leq 2n$. The third inequality follows due to $\beta_k \leq 1$ for $1 \leq k \leq K$.

It can be observed that when the power consumption function is given by $g(x) = \mu x^\alpha$, $\sum_{j_i\in\mathcal{J}} h(x_{i,e}^*(k))|I_k|$ is a lower bound for the optimal energy consumption as it is the total energy consumed when the smallest transmission rate for each flow is used and also each flow can be routed through multiple paths. Consequently, we have

$$E(\sum_{1\leq l\leq K} f(\hat{x}_e(l))|I_l|) \leq O\left(\lambda^\alpha n^{2(n-1)}\right) \cdot \Phi_{opt}, \qquad (17)$$

where the expression on the left side is the expectation of the energy consumption in the solution produced by Algorithm **Random-Schedule**. Using Markov's inequality, the probability that the energy consumption is more than $c \cdot O(n^2) \cdot \Phi_{opt}$ is no more than $1/c$. This result then can be extended to nonuniform flows by introducing an extra factor $\log^{\alpha-1} D$, which has also been shown in [16]. ■

Theorem 7. *Algorithm* ***Random-Schedule*** *can solve the DCFSR problem with the power function given in Eq. 1 while guaranteeing an approximation ratio γ of $O\left(\lambda^\alpha (n^2 \log D)^{\alpha-1}\right)$.*

Proof: We also assume power consumption function $f(x)$ for the DCFSR problem and solve it with the proposed algorithm **Random-Schedule**. In the obtained fractional solution for the multi-step F-MCF problem, we use $z_e^*(k) \in \{0, 1\}$ to indicate whether a link e is chosen or not in interval I_k. We denote by $\hat{z}_e$ an indicator in the produced solution where $\hat{z}_e = 1$ if e is used in at least one of the intervals; $\hat{z}_e = 0$ otherwise. Then, we have

$$\begin{aligned} \sigma(t_K - t_0)E(\hat{z}_e) &= \sigma(t_K - t_0)\left(1 - \prod_k (1 - E(\hat{z}_e(k)))\right) \\ &\leq \sigma(t_K - t_0) \sum_k E(\hat{z}_e(k)) \\ &= \sigma(t_K - t_0) \sum_k \left(1 - \prod_{j_i\in J} (1 - y_{i,e}^*(k))\right) \\ &\leq \sigma(t_K - t_0) \sum_k \sum_{j_i\in J} y_{i,e}^*(k) \\ &\leq \sigma(t_K - t_0) \sum_k n z_e^*(k) \qquad (18) \\ &\leq n\lambda \sum_k z_e^*(k) \sigma |I_k|, \end{aligned}$$

where the third inequality follows from $y_{i,e}^*(k) \leq z_e^*(k)$ and the last inequality follows from $\min_k |I_k| \leq |I_k|$ for any $1 \leq k \leq K$. It can be observed that $\sum_k z_e^*(k)\sigma|I_k|$ is a lower bound for the optimal idle energy consumption by link e since $z_e^*(k)$ is derived from the M-MCF problem which allows the links to be turned on and off freely at any moment. Combining all these together, we conclude that algorithm **Random-Schedule** can produce a $O\left(\lambda^\alpha (n^2 \log D)^{\alpha-1}\right)$-approximation for the DCFSR problem. ■

C. Numerical Results

We now briefly describe simulation results that illustrate the approximation performance of the proposed algorithm. We build a simulator with the **Random-Schedule** implemented in Python. We use the power consumption functions x^2 or x^4 and we choose a data center network topology which consists of 80 switches (with 128 servers connected). We consider [1, 100] as the time period of interest and as we assume no prior knowledge on the flows, we select release times and deadlines of flows randomly following a uniform distribution in [1,100]. The number of flows ranges from 40 to 200 and the amount of data from each flow is given by a random rational number following normal distribution $\mathcal{N}(10, 3)$. We compare

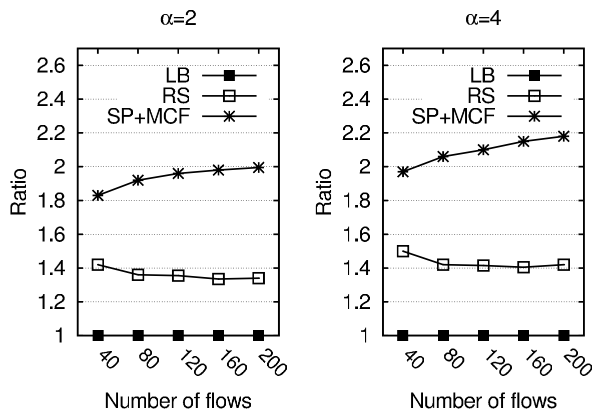

Figure 2. The approximation performance of **Random-Schedule**.

three values of interest: lower bound (LB) for the optimum (solution given by $y^*_{i,v}(k)$), **Shortest-Path** (SP) routing plus **Most-Critical-First** (MCF), and **Random-Schedule** (RS). As SP is usually adopted, SP+MCF can give the lower bound of the energy consumption by SP routing, which represents the normal energy consumption in data centers. All of the values are normalized by the lower bound for the optimum and are averaged among 10 independent runs. The simulation results are illustrated in Fig. 2. As expected, RS outperforms SP+MCF to a large extent. Moreover, we notice that with the increase of the number of flows, the approximation ratio of RS converges while the approximation ratio of SP+MCF keeps an increasing trend. This confirms that combining routing and scheduling for flows can provide substantial improvements on the energy efficiency in data center networks.

This simulation serves merely as a primitive validation of the performance of the algorithm. Due to the space limit, we leave more exhaustive evaluation and further implementation as future work.

VI. Related Work

This section summarizes some related work on the problem of improving the energy efficiency of DCNs, as well as the network scheduling and routing problem.

A. *Energy-Efficient Data Center Networks*

There has been a large body of work on improving the energy efficiency in DCNs. In general, they can be classified into two categories: The first line of work is designing new topologies that use fewer network devices while aiming to guarantee similar connectivity thus performance, such as the flatted butterfly proposed by Abts *et al.* [22] or PCube [23], a server-centric network topology for data centers, which can vary the bandwidth availability according to traffic demands. The second line of work is optimizing the energy efficiency by traffic engineering, i.e., consolidating flows and switching off unnecessary network devices. The most representative work in this category is ElasticTree [1], which is a network-wide power manager that can dynamically adjust a set of active network elements to satisfy variable data center traffic loads. Shang *et al.* [2] considered saving energy from a routing perspective, routing flows with as few network devices as possible. Mahadevan *et al.* [24] discussed how to reduce the network operational power in large-scale systems and data centers. The first rate-adaptation based solution to achieve energy efficiency for future data center networks was provided by [25]. Vasic *et al.* [4] developed a new energy saving scheme that is based on identifying and using energy-critical paths. Recently, Wang *et al.* [3] proposed CARPO, a correlation-aware power optimization algorithm that dynamically consolidates traffic flows onto a small set of links and switches and shuts down unused network devices. Zhang *et al.* [26] proposed a hierarchical model to optimize the power in DCNs and proposed some simple heuristics for the model. In [27], the authors explored the problem of improving the network energy efficiency in MapReduce systems and afterwards in [28] they proposed to improve the network energy efficiency by joint optimizing virtual machine assignment and traffic engineering in data centers. Then, this method was extended to a general framework for achieving network energy efficiency in data centers [5]. In [29], the authors proposed to combine preemptive flow scheduling and energy-aware routing to achieve better energy efficiency. But performance guarantee was not considered. In a following work [30], they considered greening data center networks by using a throughput-guaranteed power-aware routing scheme. To the best of our knowledge, the present paper is among the first to address the energy efficiency of DCNs from the aspect of scheduling and routing while guaranteeing the most critical performance criterion in DCNs - meeting flow deadlines.

B. *Network Scheduling and Routing*

There have been a few works that have investigated the problem of job scheduling with deadlines in a multi-hop network. With known injection rate, a given deadline and fixed routes, the problem of online scheduling of sessions has been investigated by [31], where they first gave a necessary condition for feasibility of sessions and gave an algorithm under which with high probability, most sessions are scheduled without violating the deadline when the necessary condition for feasibility is satisfied. In [32], the authors studied online packet scheduling with hard deadlines in general multihop communication networks. The algorithm they proposed gives the first provable competitive ratio, subject to hard deadline constraints for general network topologies. However, they didn't consider optimizing for energy efficiency.

Packet scheduling and routing for energy efficiency in general networks has also been well studied. In [16] and [33], the authors investigated to optimize the network energy efficiency from the aspect of routing and scheduling under continuous flow (transmission speed for each flow is given by a constant), by exploiting speed scaling and power-down strategy, respectively. Andrews *et al.* [34] then proposed efficient packet scheduling algorithms for achieving energy efficiency while guaranteeing network stability. In [35], they also provided efficient packet scheduling algorithms for optimizing the tradeoffs between delay, queue size and energy.

Our approach has some fundamental differences with all the aforementioned solutions. Firstly, we combine speed scaling and power-down strategies for network devices in a unified model. Secondly, we carry out optimization from the granularity of flow instead of the granularity of packet and we aim at guaranteeing flow deadlines. Lastly, we investigate the problem of achieving energy efficiency by combining both flow scheduling and routing where we have to decide not only the routing path and schedule, but also the transmission rate for each flow.

VII. Conclusions

In this paper, we studied flow scheduling and routing problem in data center networks where the deadlines of flows were strictly constrained and the objective was to minimize the energy consumption for transmitting all of the flows. The key observation in this work was that energy efficiency cannot be separately considered regardless of network performance being meeting flow deadlines the most critical requirement for it. We focused on two general versions of this problem with only scheduling and both routing and scheduling respectively. We introduced an optimal combinatorial algorithm for the version with only flow scheduling and devised an efficient approximation algorithm for the version with both routing and scheduling for flows, obtaining a provable performance ratio. With the proposed algorithms, we were able to achieve the aforementioned objective.

Acknowledgment

This research was partially supported by National Science Foundation of China (NSFC) Major International Collaboration Project 61020106002, NSFC & Hong Kong RGC Joint Project 61161160566, NSFC Project for Innovation Groups 61221062, NSFC Project grant 61202059 and 61202210, the Comunidad de Madrid grant S2009TIC-1692, and Spanish MICINN/MINECO grant TEC2011-29688-C02-01.

References

[1] B. Heller, S. Seetharaman, P. Mahadevan, Y. Yiakoumis, P. Sharma, S. Banerjee, and N. McKeown, "Elastictree: Saving energy in data center networks," in *NSDI*, 2010, pp. 249–264.

[2] Y. Shang, D. Li, and M. Xu, "Energy-aware routing in data center network," in *Green Networking*, 2010, pp. 1–8.

[3] X. Wang, Y. Yao, X. Wang, K. Lu, and Q. Cao, "Carpo: Correlation-aware power optimization in data center networks," in *INFOCOM*, 2012, pp. 1125–1133.

[4] N. Vasic, P. Bhurat, D. M. Novakovic, M. Canini, S. Shekhar, and D. Kostic, "Identifying and using energy-critical paths," in *CoNEXT*, 2011, p. 18.

[5] L. Wang, F. Zhang, J. A. Aroca, A. V. Vasilakos, K. Zheng, C. Hou, D. Li, and Z. Liu, "Greendcn: A general framework for achieving energy efficiency in data center networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 1, pp. 4–15, 2014.

[6] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "A power benchmarking framework for network devices," in *Networking*, 2009, pp. 795–808.

[7] T. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding data center traffic characteristics," *Computer Communication Review*, vol. 40, no. 1, pp. 92–99, 2010.

[8] C. Wilson, H. Ballani, T. Karagiannis, and A. I. T. Rowstron, "Better never than late: meeting deadlines in datacenter networks," in *SIGCOMM*, 2011, pp. 50–61.

[9] B. Vamanan, J. Hasan, and T. N. Vijaykumar, "Deadline-aware datacenter tcp (d2tcp)," in *SIGCOMM*, 2012, pp. 115–126.

[10] C.-Y. Hong, M. Caesar, and B. Godfrey, "Finishing flows quickly with preemptive scheduling," in *SIGCOMM*, 2012, pp. 127–138.

[11] D. Zats, T. Das, P. Mohan, D. Borthakur, and R. H. Katz, "Detail: reducing the flow completion time tail in datacenter networks," in *SIGCOMM*, 2012, pp. 139–150.

[12] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, and S. Shenker, "pfabric: Minimal near-optimal datacenter transport," in *SIGCOMM*, 2013.

[13] Z. Mao, C. E. Koksal, and N. B. Shroff, "Online packet scheduling with hard deadlines in multihop communication networks," in *INFOCOM*, 2013.

[14] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *SIGCOMM*, 2008.

[15] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "Bcube: a high performance, server-centric network architecture for modular data centers," in *SIGCOMM*, 2009, pp. 63–74.

[16] M. Andrews, A. F. Anta, L. Zhang, and W. Zhao, "Routing for power minimization in the speed scaling model," *IEEE/ACM Trans. Netw.*, vol. 20, no. 1, pp. 285–294, 2012.

[17] Y. Nesterov and A. Nemirovskii, *Interior Point Polynomial Algorithms in Convex Programming*, ser. SIAM studies in applied and numerical mathematics: Society for Industrial and Applied Mathematics. Society for Industrial and Applied Mathematics, 1994.

[18] F. F. Yao, A. J. Demers, and S. Shenker, "A scheduling model for reduced cpu energy," in *FOCS*, 1995, pp. 374–382.

[19] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness*. New York, NY, USA: W. H. Freeman & Co., 1990.

[20] V. V. Vazirani, *Approximation Algorithms*. Springer, March 2004.

[21] P. Raghavan and C. D. Tompson, "Randomized rounding: a technique for provably good algorithms and algorithmic proofs," *Combinatorica*, vol. 7, no. 4, pp. 365–374, Dec. 1987.

[22] D. Abts, M. R. Marty, P. M. Wells, P. Klausler, and H. Liu, "Energy proportional datacenter networks," in *ISCA*, 2010, pp. 338–347.

[23] L. Huang, Q. Jia, X. Wang, S. Yang, and B. Li, "Pcube: Improving power efficiency in data center networks," in *IEEE CLOUD*, 2011, pp. 65–72.

[24] P. Mahadevan, S. Banerjee, P. Sharma, A. Shah, and P. Ranganathan, "On energy efficiency for enterprise and data center networks," *IEEE Communications Magazine*, vol. 49, no. 8, pp. 94–100, 2011.

[25] L. Wang, F. Zhang, C. Hou, J. A. Aroca, and Z. Liu, "Incorporating rate adaptation into green networking for future data centers," in *NCA*, 2013, pp. 106–109.

[26] Y. Zhang and N. Ansari, "Hero: Hierarchical energy optimization for data center networks," in *ICC*, 2012, pp. 2924–2928.

[27] L. Wang, F. Zhang, and Z. Liu, "Improving the network energy efficiency in mapreduce systems," in *ICCCN*, 2013, pp. 1–7.

[28] L. Wang, F. Zhang, A. V. Vasilakos, C. Hou, and Z. Liu, "Joint virtual machine assignment and traffic engineering for green data center networks," *SIGMETRICS Performance Evaluation Review*, vol. 41, no. 3, pp. 107–112, 2013.

[29] Y. Shang, D. Li, and M. Xu, "Greening data center networks with flow preemption and energy-aware routing," in *Local Metropolitan Area Networks (LANMAN), 2013 19th IEEE Workshop on*, 2013, pp. 1–6.

[30] M. Xu, Y. Shang, D. Li, and X. Wang, "Greening data center networks with throughput-guaranteed power-aware routing," *Computer Networks*, vol. 57, no. 15, pp. 2880–2899, 2013.

[31] M. Andrews and L. Zhang, "Packet routing with arbitrary end-to-end delay requirements," in *STOC*, 1999, pp. 557–565.

[32] Z. Mao, C. E. Koksal, and N. B. Shroff, "Online packet scheduling with hard deadlines in multihop communication networks," in *INFOCOM*, 2013, pp. 2463–2471.

[33] M. Andrews, A. F. Anta, L. Zhang, and W. Zhao, "Routing and scheduling for energy and delay minimization in the powerdown model," *Networks*, vol. 61, no. 3, pp. 226–237, 2013.

[34] M. Andrews, S. Antonakopoulos, and L. Zhang, "Energy-aware scheduling algorithms for network stability," in *INFOCOM*, 2011, pp. 1359–1367.

[35] M. Andrews and L. Zhang, "Scheduling algorithms for optimizing the tradeoffs between delay, queue size and energy," in *CISS*, 2012, pp. 1–6.