Advance Bandwidth Reservation for Energy Efficiency in High-performance Networks

Tong Shu, Chase Qishi Wu, and Daqing Yun
Department of Computer Science
The University of Memphis
Memphis, TN 38152, USA
Email:{tshu1, qishiwu, dyun}@memphis.edu

***Abstract*—An increasing number of high-performance networks provision dedicated channels through circuit-switching or MPLS/GMPLS tunneling techniques to support large data transfer. The link bandwidths of these networks are typically shared by multiple users through advance scheduling and reservation. The sheer volume of data transfer across such networks in a national or international scope requires a significant amount of energy on a daily basis. However, most existing bandwidth scheduling algorithms only concern traditional objectives such as data transfer time minimization, and very limited efforts have been devoted to energy efficiency in high-performance networks. In this paper, we adopt a practical power model and formulate an advance instant bandwidth scheduling problem to minimize energy consumption under a data transfer deadline constraint. We design a polynomial-time optimal solution to this problem and provide a rigorous correctness proof. The performance superiority of the proposed solution in terms of energy saving is illustrated by extensive results based on both simulated and real-life networks in comparison with existing methods.**

I. Introduction

Data centers composed of computer servers, storage systems, and network devices have been rapidly developed and deployed across the nation and around the globe. A large data center with industrial scale operations may use as much electricity as a small town. Particularly, the data centers in the United States consumed about 1.5% of national electricity in 2006 [19]. Over the past several years, the energy used by these centers and their supporting infrastructure is estimated to have increased by nearly 100 percent.

Data centers are built on both high-performance computing (HPC) facilities and high-performance network (HPN) infrastructures. The large volumes of data processed or generated by HPC facilities are typically carried by HPN with the capability of bandwidth provisioning to support remote tasks in many data-intensive applications in various science, engineering and business domains. Several HPN projects are currently underway, including User Controlled Light Paths (UCLP) [1], Enlightened [5], Dynamic Resource Allocation via GMPLS Optical Networks (DRAGON) [2], On-demand Secure Circuits and Advance Reservation System (OSCARS) [3] of ESnet, Interoperable On-demand Network (ION) of Internet2 [4] and Bandwidth Brokers [24]. The network infrastructures such as edge devices, core switches, and backbone routers in HPN are generally coordinated by a management framework, namely control plane, which is responsible for reserving link bandwidths, setting up end-to-end network paths, and releasing resources when tasks are completed. As the central function unit of a generalized control plane, the bandwidth scheduler computes appropriate network paths and allocates link bandwidths to meet specific user requests based on network topology and bandwidth availability.

There have been substantial research efforts on various aspects of energy efficiency or power awareness for HPC systems. However, energy consideration in HPN especially for bandwidth scheduling is still very limited. Most existing bandwidth scheduling algorithms only concern traditional optimization objectives such as minimizing data transfer end time. In this paper, we adopt a practical power model to calculate the energy consumption in HPN and formulate an advance instant bandwidth scheduling problem to minimize energy consumption under a data transfer deadline constraint. We design a polynomial-time optimal solution to this problem and provide a rigorous correctness proof. The performance superiority of the proposed solution in terms of energy saving is illustrated by extensive results based on both simulated and real-life networks in comparison with existing methods.

The rest of the paper is organized as follows. Section II provides a survey of related work. Section III formulates the scheduling problem. Section IV designs the optimal solution and Section V presents the performance evaluation results.

II. Related Work

A. Green Networking

Green networking techniques fall in four categories: resource consolidation, energy-proportional computing, selective connectedness, and virtualization. The first two techniques are more suited for network infrastructures. Selective connectedness allows unused resources at the edge of a network to be shut down for energy saving. A typical example of virtualization is to share servers in data centers, thus reducing hardware and cooling costs and improving energy management [8].

Resource consolidation reduces energy waste due to over-provisioning and over-dimensioning of network infrastructures [8], e.g. by turning off some lightly loaded routers and rerouting the network traffic on a selected set of active network equipments [9]. It is often referred to as a powering-down strategy and several such methods have been proposed. Zhang *et al.* proposed an intra-domain traffic engineering mechanism, GreenTE, to maximize the number of idle links under given link utilization and packet delay constraints [23].

978-1-4799-0537-9/13/$31.00 ©2013 IEEE

Andrews *et al.* studied a periodic scheduling problem to determine the path of each traffic stream for a given network and traffic matrix [7]. They proposed a schedule to minimize the active period per network element for a line topology, which was extended to an arbitrary topology by network partition, and designed a logarithmic approximation algorithm for both energy and delay minimization. Considering that many links in core networks are actually bundles of multiple physical cables and line cards that can be shut down independently, Fisher *et al.* identified an NP-complete problem of maximizing the number of shutdown cables, and proposed several heuristics based on linear optimization techniques [12]. Chiaraviglio *et al.* investigated a way to support network traffic on a minimal subset of network resources by turning of network nodes and links under full connectivity and maximum link utilization constraints [11].

Energy-proportional computing ensures that the power consumption scales proportionally with the amount of workload [9]. Typical examples include dynamic voltage and frequency scaling and adaptive link rate [8]. In the network field, it can be viewed as a speed-scaling strategy, and several such methods have been proposed for the Internet. Tang *et al.* formulated a flow allocation problem for the cases of a single (SF-RAP) session and multiple (MF-RAP) sessions in wired networks [20]: given a set of candidate paths for each end-to-end communication session, find a feasible flow allocation to minimize the incremental power consumption, subject to the constraint that the traffic demand of each session is satisfied. Since the problem in both cases is NP-hard, they proposed a Mixed Integer Linear Programming formulation and an LP-based heuristic algorithm for MF-RAP, and designed a 2-approximation algorithm for SF-RAP. Andrews *et al.* formulated a min-power routing problem and considered various speed-power curves as a function of the processing speed [6]. When the function is superadditive, they showed that there is no bounded approximation in general for single-path routing, which is in contrast with the well-known logarithmic approximation for subadditive functions. For polynomial speed-power curves, they showed a constant approximation via a simple scheme of randomized rounding.

Most research efforts on green networking focus on the packet delivery via the IP protocol on the Internet. Similar efforts via MPLS and RSVP-TE protocols in dedicated networks are still quite limited. In this paper, we utilize both powering-down and speed-scaling strategies to achieve energy efficiency in high-performance networks.

B. Bandwidth Scheduling

As dedicated networks are increasingly developed and deployed under different high-performance networking initiatives, many scheduling algorithms have been designed for advance bandwidth reservation.

In [18], Rao *et al.* described four basic scheduling problems with different constraints on target bandwidths and time slots, i.e. specified bandwidth in a specified time slot, earliest available time with a specified bandwidth and duration, highest available bandwidth in a specified time slot, and all available time slots with a specified bandwidth and duration. The solutions to the first three problems are straightforward extensions of the classical Dijkstra's algorithm, while the last one is based on an extension of Bellman-Ford algorithm. Guerin *et al.* investigated these basic scheduling problems with several extensions in [16] with a focus on increasing the flexibility of services. In [15], Grimmell *et al.* formulated a dynamic quickest path problem, which deals with the transmission of a message from a source to a destination with the minimum end-to-end delay over a network with propagation delays and dynamic link bandwidth constraints. In [22], files are transferred with varying bandwidths in different time slots in a simple case where the path is pre-specified. Ganguly *et al.* generalized the problems of finding an optimal path in a graph with varying bandwidths to minimize the total transfer time in [13], where they also proposed to find the minimum number of path switchings for a file transfer in a specified number of time slots. In [14], Gorinsky *et al.* proposed a Virtual Finish Time First algorithm to schedule incoming files in a preemptive manner to minimize total transfer end time on a dedicated channel.

In view of different transport constraints and application requirements, Lin *et al.* formulated four types of instant bandwidth scheduling problems as follows [17]: Given a network graph with an available time-bandwidth (ATB) table combining the reservation information on all links, source v_s and destination v_d, data size δ,

- FPFB: compute a fixed path from v_s to v_d with a constant (fixed) bandwidth;
- FPVB: compute a fixed path from v_s to v_d with varying bandwidths across multiple time slots;
- VPFB: compute a set of paths from v_s to v_d with the same (fixed) bandwidth at different time slots;
- VPVB: compute a set of paths from v_s to v_d with varying bandwidths at different time slots,

with the common goal to minimize the data transfer end time.

Since many real-life HPN employ FPFB as the primary service model, our work focuses on bandwidth scheduling in the case of FPFB for energy efficiency.

III. Problem Formulation

We consider an HPN $G(V, L)$ that consists of a set V of routers connected through a set L of full duplex wired links of capacities C_L. Each router v is equipped with N_v^{LC} line cards c_i^v, $i = 1, 2, ..., N_v^{LC}$, each of which contains multiple ports. The set V of routers and the set C of line cards on all the routers make up a set D of network devices, i.e. $D = V \cup C$. A user data transfer request $R(v_s, v_d, \delta, t^A)$ specifies the source v_s, the destination v_d, the data size δ, and the time point t^A ($t^A \geq 0$) when the data are available for transfer.

We use a boot-up time (BUT) list T_D^{BU} to store the amount of time required for activating all network devices. Based on T_D^{BU} and the current bandwidth reservation status, an energy-aware bandwidth scheduler is able to shut down idle network devices for energy saving as long as the idle time is longer

than the boot-up (or activation) time. We also use a up-down state (UDS) table $U_D(t)$ to keep track of the time-varying up-down states of all network devices. If router v and line card c_i^v are powered off at time point t, $U_v(t)$ and $U(c_i^v,t)$ are set to 0; otherwise, they are set to 1.

In addition, the scheduler maintains a powered-on time (POT) list T_D^{on} to record the amount of time, during which a network device is continuously powered on up to the current time. Here, $T^{on}(c_i^v)=0$, if $T_v^{on} \le T_v^{BU}$. Based on the BUT and POT lists, the scheduler calculates an available state table $A_D(t)$ as follows: if router v or line card c_i^v is operable at time point t, $A_v(t)$ or $A(c_i^v,t)$ is set to 1; otherwise (either shut down or being booted up), it is set to 0, i.e.

$$A_v(t) = \begin{cases} 1, \text{ if } t+T_v^{on} \ge T_v^{BU}, \\ 0, \text{ if } t+T_v^{on} < T_v^{BU}. \end{cases} \quad (1)$$

$$A(c_i^v,t) = \begin{cases} 1, \text{ if } t+T_v^{on} \ge T^{BU}(c_i^v)+T_v^{BU},\ T_v^{on} \le T_v^{BU}, \\ 1, \text{ if } t+T^{on}(c_i^v) \ge T^{BU}(c_i^v),\ T_v^{on} > T_v^{BU}, \\ 0, \text{ others.} \end{cases} \quad (2)$$

The actual capacity of link l at time point t depends on the status of the routers and line cards on both ends:

$$C_l(t) = C_l \cdot \prod_{c_i^v \in l} A(c_i^v,t). \quad (3)$$

The scheduler maintains an available time-bandwidth (ATB) table $B_L^A(t)$ for all directed links L in each time slot in future time. For each directed link l, there is a step function in terms of time t to describe its available bandwidth. Once the scheduler accepts a new user request over a computed network path for a certain time duration, or a network device is shut down or booted up, the available time-bandwidth is dynamically updated to $B_l^A(t) = C_l(t) - B_l^R(t) \ge 0$, where $B_l^R(t)$ is the reserved bandwidth on link l at time point t. We denote the available time-bandwidth table from t^A as $(t[0], t[1], b_0[0], b_1[0], ..., b_{m-1}[0]), ..., (t[T'_A-1], t[T'_A], b_0[T'_A-1], b_1[T'_A-1], ..., b_{m-1}[T'_A-1])$, where T'_A is the total number of new time slots from t^A after the aggregation of the ATB table of all m links.

We use $B_p^{BN}(t)$ to denote the bottleneck (BN) bandwidth of path $p_{s,d}$ from source v_s to destination v_d at time point t:

$$B_p^{BN}(t) = \min_{l \in p} B_l^A(t). \quad (4)$$

The total data rate of all the flows via line card c_i^v at time point t is:

$$r(c_i^v,t) = \sum_{l \in L^{out}(c_i^v)} B_l^R(t) + \sum_{l \in L^{in}(c_i^v)} B_l^R(t). \quad (5)$$

We employ a general power model $P_v(r)$ to calculate the power consumption of router v with traffic load r [10]:

$$P_v(r) = P_v^S + \sum_{i=1}^{N_v^{LC}} (P^S(c_i^v) + P^D(c_i^v, r)), \quad (6)$$

where P_v^S is the static power consumption for the chassis of router v; N_v^{LC} is the number of line cards on router v; $P^S(c_i^v)$ is the static power consumption of the i-th line card on router v in a base configuration; $P^D(c_i^v, r)$ is the dynamic power consumption of line card c_i^v, which is a function of traffic load r on line card c_i^v. Hence, the energy consumption $E_p(t_1,t_2)$ incurred by a user request over path p during a time range from time t_1 to time t_2 consists of a static part $E_p^S(t_1,t_2)$ and a dynamic part $E_p^D(t_1,t_2)$:

$$E_p(t_1,t_2) = E_p^S(t_1,t_2) + E_p^D(t_1,t_2), \quad (7)$$

where

$$E_p^S(t_1,t_2) = \sum_{v \in p} \int_{t_1 - T_v^{BU} - \max_{c_i^v \in p} T^{BU}(c_i^v)}^{t_2} (1-U_v(t)) P_v^S + \sum_{c_i^v \in p} \int_{t_1 - T^{BU}(c_i^v)}^{t_2} (1-U(c_i^v,t)) P^S(c_i^v), \quad (8)$$

and

$$E_p^D(t_1,t_2) = \sum_{c_i^v \in p} \int_{t_1}^{t_2} P^D(c_i^v, r_p(t_1,t_2) \cdot n_p(c_i^v) + r(c_i^v,t)) - P^D(c_i^v, r(c_i^v,t)). \quad (9)$$

Eq. 8 calculates the incremental static energy consumption used for booting up and powering on all the necessary routers and line cards to meet the new user request. Eq. 9 calculates the incremental dynamic energy consumption of all the routers and line cards on a path. We tabulate the main notations used in the cost models in Table I for convenient reference.

Based on the above cost models, we formulate the instant scheduling problem as follows:

Definition 1: FPFB-MEC: Given a directed network graph $G(V,L)$ with link capacities C_L, a user request $R(v_s, v_d, \delta, t^A)$, a deadline t^D, an available bandwidth-time table $B_L^A(t)$, a device up-down state table $U_D(t)$, a boot-up time list T_D^{BU} as well as the power models P_D^S and $P_{LC}^D(r)$ of a linear function starting from the origin of the coordinates, we wish to find a triplet (p, t_1, t_2) of a fixed path p with a fixed data rate, start time t_1, and end time t_2 to meet the user request R with the minimum energy consumption:

$$\min_{t^A \le t_1 < t_2 \le t^D, p \in P_{s,d}} E_p(t_1,t_2), \quad (10)$$

subject to

$$(t_2 - t_1) \cdot r_p(t_1,t_2) = \delta, \quad (11)$$

$$r_p(t_1,t_2) \le \min_{t_1 \le t \le t_2} B_p^{BN}(t). \quad (12)$$

IV. Algorithm Design

A. Optimal Algorithm for FPFB-MEC

We propose an optimal algorithm for the FPFB-MEC problem, referred to as Smart Advance reserVation for Energy Efficiency (SAVEE). The pseudocode of SAVEE is provided in Alg. 1. Given a user request, the scheduler first updates the ATB table according to the BUT and POT lists (Line 1). T_{UDS} contains the start and end time points of all the time slots in the UDS table (Line 2); and T_{BU} contains the boot-up time of all the line cards when the router is powered on or shut down (Line 3). Since the data transfer must start and finish within the period from t^A to t^D, the algorithm varies the transfer start time slot x from 0 to $y-1$ for a given data transfer end time slot $y-1$, and finds the path ρ with the minimum energy consumption such that the data of size δ can be transferred during the time slot range $[x, y-1]$. The algorithm repeatedly increases y by 1, and computes the optimal transfer start time t_1 and end time t_2 by considering all possible x and y values (Lines 5-6). SAVEE further defines the following notations:

TABLE I
NOTATIONS USED IN THE PROBLEM FORMULATION.

Parameters	Definitions
$G(V, L)$	A directed network graph of a set V of routers and a set L of directed links among them
C_L	A set of the capacities of all the directed links
$C_l(t)$	The capacity of directed link l at time t
LC	A set of line cards
c_i^v	The i-th line card on router v
D	A set of network devices (routers and line cards)
$R(v_s, v_d, \delta, t^A)$	A user request for transferring data of size δ from source v_s to destination v_d after available time t^A
t^D	The deadline of data transfer
$P_{s,d}$	A set of paths from source v_s to destination v_d
$B_l^R(t)$	Reserved bandwidth of directed link l at time t
$B_l^A(t)$	Available bandwidth of directed link l at time t
$B_p^{BN}(t)$	Bottleneck bandwidth of path p at time t
$L^{out}(c_i^v)$	A set of outgoing directed links from line card c_i^v
$L^{in}(c_i^v)$	A set of incoming directed links to line card c_i^v
$r(c_i^v, t)$	The total data rate of flows on line card c_i^v at time t
$r_p(t_1, t_2)$	The data rate on path p from time t_1 to time t_2
$n_p(c_i^v)$	The number of times a flow goes through the same line card c_i^v along path p
T_D^{BU}	A boot-up time list of all the devices
T_D^{on}	A powered-on time list of all the devices
$U_D(t)$	A up-down state table of all the devices
$A_D(t)$	A available state table of all the devices
N_v^{LC}	The number of line cards on router v
$P_v(r)$	Power consumption of router v with traffic load r
P_v^S	Static power consumption of the chassis of router v
$P^S(c_i^v)$	Static power consumption of line card c_i^v
$P^D(c_i^v, r)$	Dynamic power consumption of line card c_i^v with traffic load r
$E_p(t_1, t_2)$	Incremental energy consumption over path p from time t_1 to time t_2
$E_p^S(t_1, t_2)$	Incremental static energy consumption over path p from time t_1 to time t_2
$E_p^D(t_1, t_2)$	Incremental dynamic energy consumption over path p from time t_1 to time t_2

- b_l: the maximum available bandwidth of link l over the entire time slot range $[x, y-1]$ (Lines 7-8),
- B_0: the minimum bandwidth to transfer the data of size δ during time slots $[x, y-1]$ (Line 9),
- B_1: the maximum bandwidth to transfer the data of size δ from the beginning of start time slot x to end time slot $y-1$ (Line 10),
- B_2: the maximum bandwidth to transfer the data of size δ from start time slot x to the end of end time slot $y-1$ (Line 20).

In each time range, SAVEE selects all the link bandwidths within the upper and lower boundaries as mentioned above (Lines 11 and 22). For a given path and a given bandwidth, the optimal bandwidth reservation either starts at time $t_s \in T_S$ (T_S is the set of all possible data transfer start time points as defined in Line 12) or ends at time $t_e \in T_E$ (T_E is the set of all possible data transfer end time points as defined in Line 22), depending on the static energy consumption of data transfer over different time slots in the UDS table (Lines 12 and 23), and thus SAVEE calculates the exact time range $[\tau_1, \tau_2]$ (Lines 14-15, 24-25). Then, the minimum energy consumption ε in this time range and the corresponding path are calculated using Dijkstra's algorithm based on a new graph constructed from the original one (Lines 16-17, 26-27). SAVEE guarantees that the returned energy consumption is minimized since it examines all possible transfer time ranges and bandwidths (Lines 11-19, 21-29). Finally, SAVEE updates the ATB table to reserve the bandwidth, and the UDS table to boot up necessary devices and shut down unused devices (Line 30).

Algorithm 1 $SAVEE$

Input: G, C_L, R, T_D, ATB, UDS, BUT, POT and PM

Output: the minimum energy consumption $E_{\min}$, path p, start time t_1 and end time t_2

1: Updates the ATB table according to the BUT and POT lists;
2: $T_{UDS} = \{$the start and end time points of all the time slots in the UDS table$\}$;
3: $T_{BU} = \{0, T^{BU}(c_i^v), T^{BU}(c_i^v) + T_v^{BU} | v \in V, c_i^v \in LC\}$;
4: $E_{\min} = \infty$;
5: **for** $y = 1$ to T'_A **do**
6: **for** $x = 0$ to $y-1$ **do**
7: **for all** $l \in L$ **do**
8: $b_l = \min_{x \le i \le y-1} b_l[i]$;
9: $B_0 = \frac{\delta}{t[y]-t[x]}$;
10: $B_1 = \frac{\delta}{t[y-1]-t[x]}$;
11: **for all** $\beta \in \{b_l | B_0 \le b_l < B_1, l \in L\}$ **do**
12: $T_S = \{t'+t'' | t[x] \le t'+t'' \le t[y] - \frac{\delta}{\beta}, t' \in T_{UDS}, t'' \in T_{BU}\}$;
13: **for all** $t_s \in T_S$ **do**
14: $\tau_1 = t_s$;
15: $\tau_2 = t_s + \frac{\delta}{\beta}$;
16: Construct a weighted directed graph $G'(V', L')$, where $V' = \{l \in L | b_l \ge \beta, l$ is neither from v_d nor to $v_s\} \cup \{l_s, l_d\}$, $L' = \{(l_1, l_2) | l_1$ and l_2 are incoming and outgoing links of router v, respectively, but l_1 and l_2 are not the bidirectional links between the same pair of routers, $l_1, l_2 \in V'\}$ and $W_{L'} = \{E(l', \delta, \tau_1, \tau_2), l' \in L'\}$;
17: $(\varepsilon, \rho) =$ the minimum energy cost and the corresponding path to transfer the data of size δ from l_s to l_d during the exact time range $[\tau_1, \tau_2]$ based on G';
18: **if** $\varepsilon < E_{\min}$ **then**
19: $E_{\min} = \varepsilon$; $p = \rho$; $t_1 = \tau_1$; $t_2 = \tau_2$;
20: $B_2 = \frac{\delta}{t[y]-t[x+1]}$;
21: **for all** $\beta \in \{b_l | B_0 \le b_l < B_2, l \in L\}$ **do**
22: $T_E = \{t' | t[x] + \frac{\delta}{\beta} \le t' \le t[y], t' \in T_{UDS}\}$;
23: **for all** $t_e \in T_E$ **do**
24: $\tau_1 = t_e - \frac{\delta}{\beta}$;
25: $\tau_2 = t_e$;
26: Construct a weighted directed graph $G'(V', L')$ with weights of $W_{L'}$ in the same way as above;
27: Compute the minimum energy consumption and the corresponding path (ε, ρ);
28: **if** $\varepsilon < E_{\min}$ **then**
29: $E_{\min} = \varepsilon$; $p = \rho$; $t_1 = \tau_1$; $t_2 = \tau_2$;
30: Update the ATB table and the UDS table to indicate when the routers and line cards are booted up or shut down.
31: **return** $(E_{\min}, p, t_1, t_2)$.

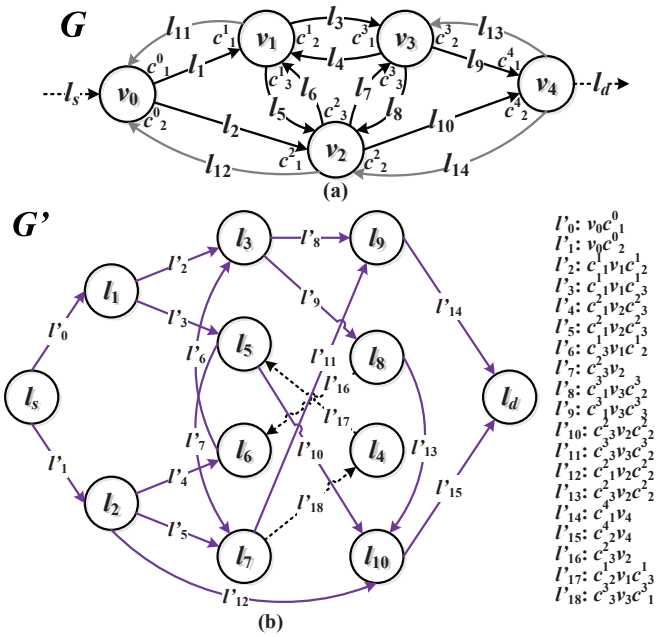

Fig. 1. A small network example: (a) the original network graph; (b) the newly constructed virtual graph.

In the original network graph, the number of all possible paths is exponential, which prohibits an exhaustive search. Since routers and line cards are the main energy consumers, we assign each router with ingress and egress line cards a weight of energy consumption (not a link weight). Therefore, the shortest path algorithm does not work on the original network graph. To address this issue, we construct a new weighted directed graph $G'(V', L')$ with weights of $W_{L'}$. Here, the set of vertices $V' = \{l \in L | b_l \geq \beta, l$ is neither an outgoing link of v_d nor an incoming link of $v_s\} \cup \{l_s, l_d\}$, where l_s is a virtual link to source v_s and l_d is a virtual link from destination v_d in G. The set L' contains pairs of links in G that are connected without forming a loop, and each edge l' can be represented by a triplet of a router v and its ingress and egress line cards c_i^v, c_e^v that connect the pair of directed links in G. The weight $W_{l'}$ of edge $l' \in L'$ is the energy consumption of the triplet (v, c_i^v, c_e^v) for transferring data of size δ from time τ_1 to time τ_2, which is a constant for a given data rate within a given time range. For illustration purposes, we provide a small network example in Fig. 1, where $V = \{v_0, v_1, v_2, v_3, v_4\}$, $L = \{l_1, l_2, \ldots, l_{14}\}$, $v_s = v_0$ and $v_d = v_4$, and the line card configurations are as follows: v_0 has line cards c_1^0 and c_2^0, v_1 has line cards c_1^1, c_2^1 and c_3^1, v_2 has line cards c_1^2, c_2^2 and c_3^2, v_3 has line cards c_1^3, c_2^3 and c_3^3, and v_4 has line cards c_1^4 and c_2^4. In the new graph constructed from the original one, $V' = \{l_1, l_2, \ldots, l_{10}, l_s, l_d\}$ and L' is shown in Fig. 1(b).

B. The Correctness Proof for SAVEE

Lemma 1: Given the start or end time of a data transfer, the optimal reserved bandwidth is in the set of available bandwidths of all the links in certain time ranges.

Proof: Given a data size δ to be transferred on any fixed path, energy consumption $E_p(\delta) = E_p^S(\delta) + E_p^D(\delta) = E_p^S(r \cdot t) + \delta \cdot (P_p^D(r) \cdot t)/(r \cdot t) = E_p^S(r \cdot t) + \delta \cdot P_p^D(r)/r$. Hence, minimizing energy consumption $E_p(\delta)$ is equivalent to minimizing both $E_p^S(r \cdot t)$ and $P_p^D(r)/r$. The dynamic power consumption per data rate is calculated as follows:

$$\frac{d(P_p^D(r)/r)}{dr} = \frac{1}{r}\left(\frac{dP_p^D(r)}{dr} - \frac{P_p^D(r)}{r}\right). \quad (13)$$

Since $P_p^D(r)$ is a concave function with $P_p^D(0) = 0$, $P_p^D(r - \Delta r) \geq P_p^D(r) \cdot (r - \Delta r)/r$ $(\Delta r \geq 0)$. Thus, we have $(P_p^D(r) - P_p^D(r - \Delta r))/\Delta r \leq P_p^D(r)/r$. Then, we have

$$\frac{dP_p^D(r)}{dr} = \lim_{\Delta r \to 0} \frac{P_p^D(r) - P_p^D(r - \Delta r)}{\Delta r} \leq \frac{P_p^D(r)}{r}. \quad (14)$$

Therefore, $\frac{d(P_p^D(r)/r)}{dr} \leq 0$. In addition, $E_p^S(r \cdot t)$ is a monotonically increasing function of t and thus a monotonically decreasing function of r for the given data size δ. That is, the energy consumption $E_p(\delta)$ is minimized when r achieves the maximum available bandwidth $B_p^A(T)$ of path p during the shortest period T under the constraint $T \cdot B_p^A(T) = \delta$. Since $B_p^A(T) = \min_{0 \leq t \leq T} \min_{l \in p} B_l^A(t) \in \{B_l^A(t) | l \in L, 0 \leq t \leq T\}$, the reserved bandwidth of the optimal solution is in the set of available bandwidths of all the links in certain time ranges. Proof ends. ■

Lemma 2: The shortest path in G' represents a path with the minimum energy consumption for the data transfer from source v_s to destination v_d at rate r in the time slot from time t_1 to time t_2 in network G.

Proof: Given a triplet (r, t_1, t_2) of data rate, start time, and end time, we generate a network graph $G''(V, L'')$, where $L'' = \{l | l \in L$ and $B_l^A(t) \geq r, \forall t_1 \leq t \leq t_2\}$. The network G'' includes all the feasible paths for the data transfer from v_s to v_d at rate r from t_1 to t_2 in G. The path with the minimum energy consumption for the data transfer from v_s to v_d at rate r from t_1 to t_2 must be a simple path in G''. According to the construction method, any simple path in G'' corresponds to the only simple path in G', whose weight is the energy consumption for the data transfer via the path from v_s to v_d at rate r from t_1 to t_2. Any simple path p_0' in G' corresponds to the only path p_0'' in G''. If p_0'' is not a simple path, p_0' is not the shortest path in G', because deleting the circles in the path p_0'' can produce a simple path p_1'' that corresponds to a path p_1' in G' shorter than p_0'. Therefore, the shortest path in G' corresponds to a single path with the minimum energy consumption for the data transfer from v_s to v_d at rate r from t_1 to t_2 in network G. Proof ends. ■

We use t_S and t_E to denote the start and end time of one or multiple continuous time slots in the ATB table $(t_S < t_E)$, and use $B_L^A(t_S, t_E)$ to denote the set of available bandwidths of all the links from time t_S to time t_E. $T_S(t_S, t_E) = \{t' + t'' | t_S \leq t' + t'' \leq t_E - \frac{\delta}{\beta}, t' \in T_{UDS}, t'' \in T_{BU}, \beta \in B_L^A(t_S, t_E)\}$, and $T_E(t_S, t_E) = \{t' | t_S + \frac{\delta}{\beta} \leq t' \leq t_E, t' \in T_{UDS}, \beta \in B_L^A(t_S, t_E)\}$. We have the following lemma:

Lemma 3: Given a data transfer request, there exists an optimal bandwidth reservation, which either starts at a time point in T_S or ends at a time point in T_E.

Proof: Since we consider a linear dynamic power model with respect to the data rate, the dynamic energy consumption of data transfer only depends on the size of data to be transferred on any fixed path. In addition, the static power consumption of data transfer remains the same within one time slot in the UDS table on any fixed path. According to Lemma 1, $B_L^A(t_S, t_E)$ contains all possible bandwidths of the optimal bandwidth reservation from t_S to t_E. The data transfer within the time range from t_S to t_E at rate β must start before $t_E - \frac{\delta}{\beta}$ and must end after $t_S + \frac{\delta}{\beta}$. Since there exists an optimal bandwidth reservation that either ends at the end of a time slot in the UDS table or immediately follows a boot-up period in T_{BU} (defined in Line 3 in Alg. 1) that begins at the start time of a time slot in the UDS table, there exists an optimal bandwidth reservation that either starts at a time point in T_S or ends at a time point in T_E. Proof ends. ■

Theorem 1: The SAVEE algorithm yields an optimal solution to the FPFB-MEC problem.

Proof: Given a data size and a reserved bandwidth, the data transfer time is a constant. Therefore, according to Lemma 1 and 3, SAVEE tries all possible triplets of the bandwidth and the time ranges of one optimal bandwidth reservation. According to Lemma 2, for a given bandwidth and a given time range, SAVEE computes the path with the minimum energy consumption. In sum, SAVEE finds a triplet (p, t_1, t_2) of the path, start time and end time with the minimum energy consumption to transfer the data at a fixed rate on a fixed path. Proof ends. ■

According to the ATB table, the scheduler shuts down idle devices only at the beginning of some time slots in the ATB table and boots up these devices if necessary in the future. Let T_A and T_U be the numbers of time slots in the ATB and UDS tables from the current time, respectively. T_U (i.e. $|T_{UDS}|$) is no more than $T_A \cdot (|D|+1|)$. $|T_{BU}| \leq 2|D|$, and thus $|T_S|$ is no more than $2 \cdot T_U \cdot |D|$. Similarly, $|T_E|$ is no more than T_U. The time complexity of Lines 7-8 is $O(T_A)$. The time complexity of Lines 16-17 is $O(|L|^2)$ according to Dijkstra's algorithm. The time complexity of Lines 11-19 is $O(T_U \cdot |L|^3 \cdot |D|)$. Therefore, the time complexity of the SAVEE algorithm is $O(T_A^3 \cdot |L|^3 \cdot |D|^2)$.

V. Performance Evaluation

A. Simulation Setup

Since FPFB-MEC is an instant bandwidth scheduling problem, the optimal solution of SAVEE does not automatically guarantee the optimality of the overall energy saving in HPN with continuously arriving user requests over a period of time. We conduct a simulation-based performance evaluation of SAVEE in comparison with the minimum end time (MET) algorithm, i.e. the OptFPFB algorithm in [17], and an energy-aware version of MET, referred to as EAMET. MET does not consider the energy consumption of network devices and always powers on all the routers and line cards; while EAMET shuts down idle routers and line cards whenever possible. In the simulation, the scheduler uses EAMET to find the earliest end time of data transfer, which is then used as a base point for setting an appropriate deadline constraint for SAVEE in FPFB-MEC.

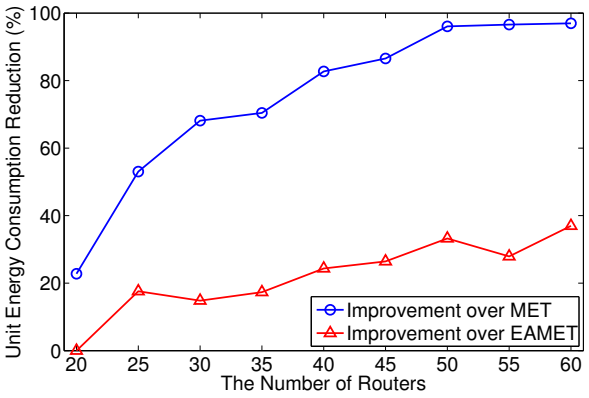

Fig. 2. Performance improvement of SAVEE in terms of UEC under different network sizes.

We investigate these algorithms in two types of networks: i) simulated random networks of 20 to 60 nodes with 10% links of a complete network, and (ii) a real-life high-performance network ESnet5. The arrivals of user requests follow the Poisson distribution. The data sizes are of the lognormal distribution mainly within 12.5 Gigabytes to 12.5 Petabytes. The parameters of the power model are chosen according to Cisco CRS-3 100G router and Cisco 7603 10G router. Each scheduling simulation lasts for 2 months and is repeated 10 times with different sources and destinations in the user requests. In the performance evaluation, each data point denotes the average result of 10 runs.

B. Random Simulated Networks

In random simulated networks, the capacity of the core link is of 100 Gbps and the capacity of the edge link is of 10 Gbps. Here, the edge link is connected to a leaf node, and the core link connects two non-leaf nodes.

1) Scalability: We run MET, EAMET, and SAVEE under different network sizes for scalability test. The average arrival interval of user requests is set to be 3 hours, and the deadline of data transfer in SAVEE is set to be the minimum end time calculated by EAMET. We define the unit energy consumption (UEC) as the ratio of the total energy consumption to the size of data transfer, and plot the UEC measurements in Fig. 2. These measurements show that SAVEE saves energy from 22% to 97% in comparison with MET and saves energy from 0 to 37% in comparison with EAMET, as the number of routers increases from 20 to 60 at an interval of 5. We also observe that the energy-saving performance improves as the network size increases, which is mainly due to the fact that i) there are more paths to choose from in larger networks and ii) paths are more likely to be merged when the network is lightly loaded.

2) Traffic Load: We further examine the performance of MET, EAMET, and SAVEE in terms of UEC under different traffic loads in a medium-sized network of 40 nodes. The deadline of data transfer in SAVEE is set to be the minimum end

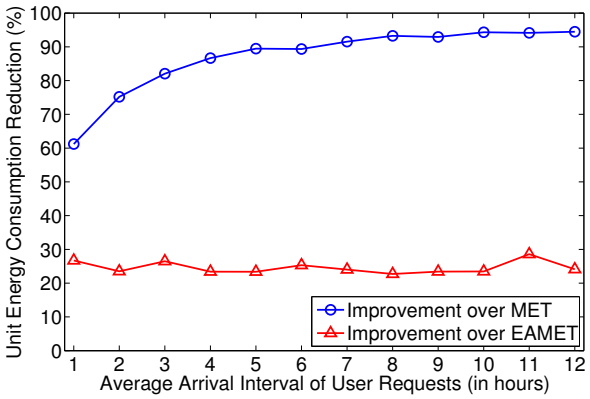

Fig. 3. Performance improvement of SAVEE in terms of UEC with different arrival intervals of user requests in random networks.

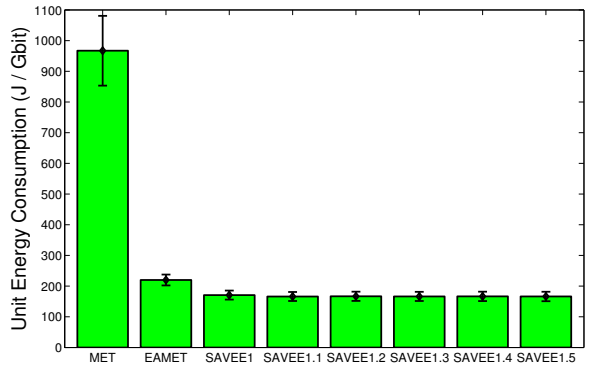

Fig. 4. UEC of SAVEE with different deadline constraints in random networks.

time calculated by EAMET. The performance measurements are plotted in Fig. 3, which shows that SAVEE saves energy from 61% to 94% in comparison with MET and saves energy around 25% in comparison with EAMET as the average arrival interval of user requests varies from 1 hour to 12 hours at an interval of 1 hour. Note that more frequent request arrivals correspond to higher traffic loads.

3) Deadline Constraints: We investigate the UEC of MET, EAMET, and SAVEE under different data transfer deadline constraints in the network of 40 nodes. The average arrival interval of user requests is set to be 3 hours. The data transfer deadline in SAVEE varies from 1 to 1.5 times of the minimum end time calculated by EAMET, at an interval of 0.1. We plot the UEC measurements together with the standard deviations in Fig. 4, which shows that SAVEE saves energy from 82% to 83% in comparison with MET and saves energy from 22% to 25% in comparison with EAMET. It is also interesting to point out that the impact of deadline constraints on the energy-saving performance of SAVEE is not very obvious. Therefore, in practice, we may choose a deadline constraint that is close to the minimum transfer end time calculated by EAMET.

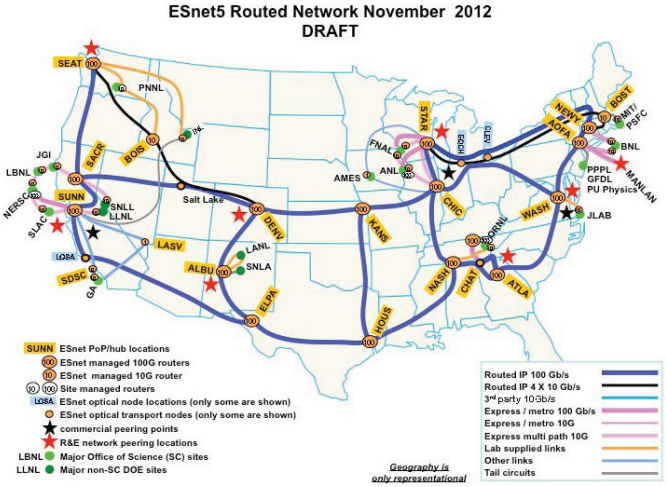

Fig. 5. The geographical layout of ESnet5 [21].

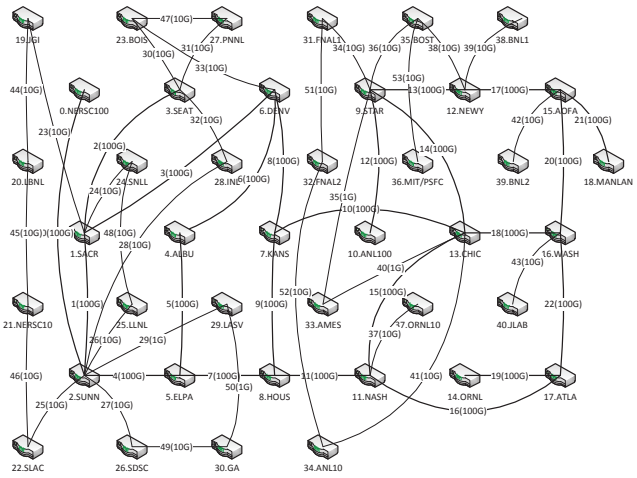

Fig. 6. The logical topology of ESnet5.

C. A Real-life Network – ESnet5

We evaluate the performance of SAVEE using a real-life high-performance network, DOE's ESnet5, whose geographical layout and logical topology are shown in Figs. 5 and 6, respectively.

1) Traffic Load: We run MET, EAMET, and SAVEE in ESnet5 with different arrival intervals of user requests. The data transfer deadline constraint is set to be 1.2 times of the minimum end time calculated by EAMET. We plot the performance measurements in Fig. 7, which shows that SAVEE saves energy from 14% to 74% in comparison with MET and saves energy from 6% to 15% in comparison with EAMET as the average arrival interval of user requests varies from 1 hour to 12 hours at an interval of 1 hour.

2) Deadline Constraints: We run MET, EAMET, and SAVEE in ESnet5 with different data transfer deadline constraints. The average arrival interval of user requests is set to be 3 hours. The deadline constraint varies from 1 to 1.5 times of the minimum end time calculated by EAMET, at an interval of

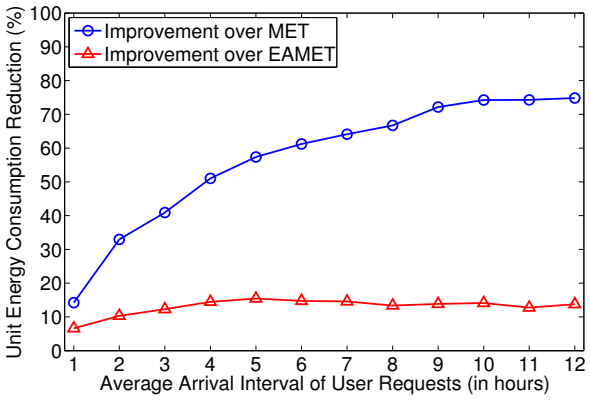

Fig. 7. Performance improvement of SAVEE in terms of UEC over different arrival intervals of user requests in ESnet.

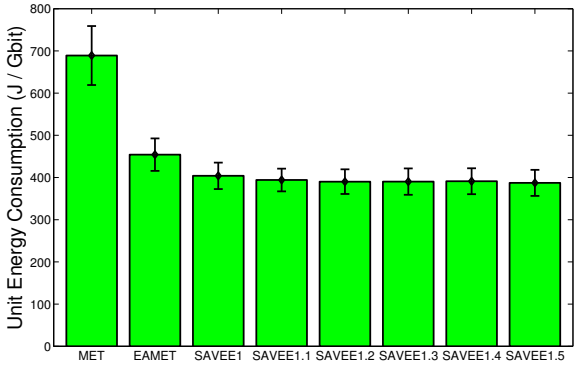

Fig. 8. UEC of SAVEE with different deadline constraints in ESnet.

0.1. We plot the UEC measurements together with the standard deviations in Fig. 8, which shows that SAVEE saves energy from 41% to 44% in comparison with MET and saves energy from 11% to 15% in comparison with EAMET. Again, we observe that the impact of deadline constraints on SAVEE's energy saving performance is not obvious, especially after the deadline constraint is extended to 1.2 times of the minimum transfer end time.

VI. Conclusion

We formulated an advance instant bandwidth scheduling problem in high-performance networks to minimize energy consumption of data transfer under a given deadline constraint. This problem is solved using an optimal algorithm with polynomial-time complexity with respect to the network size and the total number of time slots in an available time bandwidth table. Our work reveals that bandwidth scheduling that takes energy consumption into consideration could lead to significant energy saving in comparison with the existing scheduling algorithms with focus on traditional optimization objectives.

Acknowledgment

This research is sponsored by U.S. DOE's Office of Science under Grant No. DE-SC0002400 with the University of Memphis.

References

[1] UCLP: User Controlled LightPath Provisioning. http://www.uclp.ca.
[2] DRAGON: Dynamic Resource Allocation via GMPLS Optical Networks. http://dragon.maxgigapop.net.
[3] OSCARS: On-demand Secure Circuits and Advance Reservation System. http://www.es.net/oscars.
[4] Internet2 Interoperable On-Demand Network (ION) Service. http://www.internet2.edu/ion.
[5] "Enlightened computing: An architecture for co-allocating network, compute, and other grid resources for high-end applications," in *Proc. of IEEE Honet*, Dubai, UAE, Nov. 2007.
[6] M. Andrews, A.F., L. Zhang, and W. Zhao, "Routing for power minimization in the speed scaling model," *IEEE/ACM Tran. on Net.*, vol. 20, no. 1, pp. 285–294, 2012.
[7] M. Andrews, A. Anta, L. Zhang, and W. Zhao, "Routing and scheduling for energy and delay minimization in the powerdown model," in *Proc. of the 29th IEEE INFOCOM*, Piscataway, NJ, USA, Mar. 14-19 2010, pp. 21–25.
[8] A. Bianzino, C. Chaudet, D. Rossi, and J.-L. Rougier, "A survery of green networking research," *IEEE Communications Surveys and Tutorials*, vol. 14, no. 1, pp. 3–20, 2012.
[9] K. Bilal, S. Khan, S. Madani, K. Hayat, M. Khan, N. Min-Allah, J. Kolodziej, L. Wang, S. Zeadally, and D. Chen, "A survey on green communications using adaptive link rate," *J. of Cluster Computing*, 2013.
[10] J. Chabarek, J. Sommers, P. Barford, C. Estan, D. Tsiang, and S. Wright, "Power awareness in network design and routing," in *Proc. of the 27th IEEE INFOCOM*, Phoenix, AZ, USA, Apr. 13-18 2008, pp. 457–465.
[11] L. Chiaraviglio, M. Mellia, and F. Neri, "Minimizing ISP network energy cost: Formulation and solutions," *IEEE/ACM Tran. on Net.*, vol. 20, no. 2, pp. 463–376, 2012.
[12] W. Fisher, M. Suchara, and J. Rexford, "Greening backbone networks: Reducing energy consumption by shutting off cables in bundled links," in *Proc. of the 1st ACM SIGCOMM Workshop on Green Networking*, New Delhi, India, Aug. 30 2010, pp. 29–34.
[13] S. Ganguly, A. Sen, G. Xue, B. Hao, and B. Shen, "Optimal routing for fast transfer of bulk data files in time-varying networks," in *Proc. of IEEE Int. Conf. on Communications*, 2004.
[14] S. Gorinsky and N. Rao, "Dedicated channels as an optimal network support for effective transfer of massive data," in *INFOCOM Workshop on High-Speed Networks*, 2006.
[15] W. Grimmell and N. Rao, "On source-based route computation for quickest paths under dynamic bandwidth constraints," *Int. J. on Foundations of Computer Science*, vol. 14, no. 3, pp. 503–523, 2003.
[16] R. Guerin and A. Orda, "Networks with advance reservations: the routing perspective," in *Proc. of the 19th IEEE INFOCOM*, 2000.
[17] Y. Lin and Q. Wu, "Complexity analysis and algorithm design for advance bandwidth scheduling in dedicated networks," *ACM/IEEE Transactions on Networking*, vol. 21, no. 1, pp. 14–27, 2013.
[18] N. Rao, Q. Wu, S. Carter, W. Wing, D. G. A. Banerjee, and B. Mukherjee, "Control plane for advance bandwidth scheduling in ultra high-speed networks," in *INFOCOM Workshop on Terabits Networks*, 2006.
[19] J. Taheri and A. Zomaya, *Energy-Efficient Distributed Computing Systems*, A. Zomaya and Y. Lee, Eds. Wiley-IEEE Computer Society.
[20] J. Tang, B. Mumey, Y. Xing, and A. Johnson, "On exploiting flow allocation with rate adaptation for green networking," in *Proc. of the 31th IEEE INFOCOM*, Orlando, Florida, USA, Mar. 25-30 2012, pp. 1683–1691.
[21] U.S. Department of Energy. (2013, July) Esnet5 100gbps routed network map. [Online]. Available: http://www.es.net/introducing-esnet5/network-maps
[22] M. Veeraraghavan, H. Lee, E. Chong, and H. Li, "A varying-bandwidth list scheduling heuristic for file transfers," in *Proc. of IEEE Int. Conf. on Communications*, 2004.
[23] M. Zhang, C. Yi, B. Liu, and B. Zhang, "GreenTE: Power-aware traffic engineering," in *Proc. of the 18th IEEE ICNP*, Kyoto, Japan, Oct. 5-8 2010, pp. 21–30.
[24] Z. Zhang, Z. Duan, and Y. Hou, "Decoupling QoS control from core routers: A novel bandwidth broker architecture for scalable support of guaranteed services," in *Proc. of ACM SIGCOMM*, 2000.