Energy-Efficient Overlay for Data Transfers in Private Networks

Anne-Cécile Orgerie
ENS de Lyon - LIP
46 allée d'Italie 69364 Lyon Cedex 07 - France
Email: annececile.orgerie@ens-lyon.fr

Laurent Lefèvre
INRIA - LIP
46 allée d'Italie 69364 Lyon Cedex 07 - France
Email: laurent.lefevre@inria.fr

***Abstract*—Enterprise networks have been built for decades, increasing in size and complexity mostly due to richer media and the rising traffic demands of applications. Although management and other techniques have been able to cope with network complexity, little attention has been paid to curb the energy consumption. We have proposed the High-level Energy-awaRe Model for bandwidth reservation in End-to-end networkS (HERMES), a framework that ensures energy-efficiency by using scheduling algorithms that rely on aggregation of bandwidth reservations, on/off mechanisms and usage predictions. In this paper, we present new functionalities of this framework and we show through simulation results that HERMES can be used as an overlay to save substantial amounts of energy in private networks.**

I. Introduction

Private networks have been the preferred network infrastructure for large enterprises and banks due to the level of isolation and security that they provide. Over the years, however, these networks have grown larger, become more complex and, though little attention has been given to it, consumed large amounts of electricity [13].

To improve availability, networks are often over-provisioned, which typically leads to a very low overall utilization [6]. Exploring periods of network under-utilization could result in significant energy savings, and with that goal, we have proposed the High-level Energy-awaRe Model for bandwidth reservation in End-to-end networks (HERMES) [18], [17]. HERMES ensures energy-efficiency by scheduling advance bandwidth reservations using request aggregation, on/off mechanisms and usage predictions.

Although organizations can generally feel compelled to reduce their energy consumption, they might be reluctant to follow approaches that increase the burden of managing the network infrastructure. Overlay networks are an attractive solution as they provide effective service deployment, dealing with complexity without exposing it to end-users [11]. In this paper, we focus on new functionalities of HERMES that have not been described previously. We focus on private networks because they are under the control of a single administrative domain where network-wide management can be applied. [18] has presented the basic principles of an energy-efficient reservation framework without evaluation on realistic network infrastructures while [17] has quickly described HERMES' components and evaluate it on a three-tier network architecture in the context of data center networks. This paper presents the locking mechanism required by the scheduling algorithm and a re-planning capacity to increase the energy gains. It also provides new results on two new types of networks taken in the literature: an interbank network and an enterprise network.

The rest of this paper is organized as follows. Section II provides background information. HERMES architecture and the new functionalities are detailed in Section III. Section IV presents results from evaluating a realistic topology and power profiles using the Bookable Network Simulator (BoNeS). Finally, Section V concludes the paper and presents future work.

II. Background

A. Private Networks

Private networks – e.g., those used in enterprises and among banks – are dedicated infrastructure with strong requirements in terms of quality of service and security. Although this paper focuses on private networks, our model is generic enough and applicable to other types of networks.

Most traffic of banking networks is concentrated on a few links. For example, in the Fedwire network (Federal Reserve Wire Network), 1% of the nodes are responsible for 75% of the transferred data; a traffic that presents patterns and periodicity [22]. Moreover, the network core is densely connected, with a high level of redundancy. Similar observations have been made about the banking network of Austria [4] and the United-Kingdom interbank network [2].

Enterprise networks also present interesting characteristics. Most traffic is local to the enterprise and nearly all the data volume is composed of relatively large transfers, such as bulk transfers (FTP, HPSS), backup operations and network file transfers (NFS, NCP) [20].

These characteristics encourage us to use end-to-end bandwidth reservations to have a complete, and thus more efficient, control over the network, and to apply on/off mechanisms, usage predictions and traffic aggregation algorithms for energy saving purposes.

B. Bandwidth Allocation Algorithms

The idea of Advance Bandwidth Reservations (ABR) of network resources is not recent [21], but the main issue preventing its wide usage is the unpredictable routing behavior of large networks and interoperability. Different techniques can be used to schedule advance reservations for data transfers:

online scheduling where requests are processed as they arrive, and *periodic batch scheduling* where they are scheduled with certain periodicity [12]. As of writing, none of the proposed solutions takes into account the network's energy consumption as an issue that influences the design of algorithms for management, request scheduling, or packet routing.

C. Green Wired Networking

Despite the ever-increasing power consumption of wired networks [6], their energy demands can be greatly reduced by using certain techniques. Studies have shown that some links, especially those at network edges, are often lightly utilized [6], [15], a fact that led researchers to propose approaches to take advantage of link under-utilization to save energy. The first approach, known as *shutdown*, consists in switching off network equipments (or putting them in sleep mode) when they are not in use [9]. This technique raises several problems, including connectivity loss, long resynchronization time, and the fact that constantly switching equipments on and off can consume more energy than keeping them powered on. New mechanisms have been designed to minimize these issues, such as proxy techniques to maintain connectivity [14] and resynchronization of link extremities [10]. The second approach, often termed as *slowdown*, stems from the observation that NICs and switches consume less energy when operate at lower data rates [8]. This has resulted in techniques, similar to Dynamic Voltage Frequency Scaling (DVFS) for CPUs, that dynamically adjust a link's rate according to the load [8].

III. HERMES

Bandwidth brokers have long been designed for network infrastructure as a means to guarantee reliability and robustness for applications. These broker techniques maximize performance and do not consider energy consumption. Our proposed model, termed as HERMES, searches an essential trade-off between performance (volume of accepted transfers) and energy consumption. This paper provides an overview of HERMES, whereas a detailed description has been presented previously [18], [17]. We build on HERMES to propose additional features that allow it to be used as an overlay for energy-efficient reservation management.

A. HERMES architecture

The HERMES overlay is not only designed for networks deployed on large-scale distributed systems such as data centers, grids and clouds, but also for dedicated networks used in backup and e-Science. To achieve energy-efficiency, it combines several techniques built on top of existing architectures:

- unused network components are put into sleep mode;
- energy optimization of the reservation scheduling through reservation aggregation;
- minimization of the number of control messages required by the infrastructure;
- usage of Disruptive Tolerant Network (DTN) techniques to manage the infrastructure; and
- network-usage prediction to avoid frequent on/off cycles.

Each data transfer (or flow) between two nodes of the managed infrastructure requires a reservation and should first be submitted to HERMES. Then, HERMES schedules the request, informs sender (source) and receiver (destination) about the transfer schedule, and guarantees that the transfer occurs without congestion.

Each network equipment (i.e. router, switch, bridge, NIC) has two agendas per physical port (interface), one for each traffic direction (in and out). An *agenda* stores all future reservations concerning its one-way link; information that is also called the book-ahead interval [5]. A *bandwidth portion*, which can be either a fixed amount of bandwidth or a fraction of the link's capacity, is always kept available on each link for transmitting management messages and ACKs.

Furthermore, each network equipment has an agenda stating the on and off periods and the switching stages between on's and off's. In fact, this global agenda combines all equipment's port agendas. When no port is used for a certain period (not too small), the network equipment can be switched off. Usage-prediction algorithms, described later, are used to avoid switching the equipment off if it will be used in near future.

Each port maintains its reservation status using a *time-bandwidth list* (TB list) formed by $(t[i], b[i])$ tuples, where $t[i]$ is the time and $b[i]$ is the bandwidth available. Each $t[i]$ is called an *event* in the agenda.

B. The reservation process

The reservation process comprises four phases: 1) a user submits a reservation request (specifying at least data volume and required deadline) to the network-management system (detailed later); 2) the advance-reservation environment starts the negotiation phase including admission control, reservation scheduling and optimization policies; 3) a notification is sent to the user when his/her request is accepted or rejected, and another when it is scheduled; 4) the reservation starts at the scheduled start time and ends at the scheduled end time, which occurs before the user-submitted deadline.

The reservation process relies on gateways, to which end-users or servers are linked. Users, who know only their respective gateways, use a comprehensive reservation format to send their requests to a daemon running on the gateway. Each request can, in addition to a data volume to be transferred and a deadline which are mandatory, include the maximum transmission rate, a bandwidth profile (maximum available bandwidth over time for applications that generate data online with various rates) and an earliest start time (which is by default the submission time).

When a gateway receives a reservation request, the first operation that it executes is admission control, also checking whether the request is valid. Then, for each request the gateway gathers all agendas from all equipments (ports and routers) along the network path between the source and the destination. All agendas in the path are sent to the receiver's gateway during the agenda collection process. The sender gateway sends a specific management message called *control message*. Each node node that receives the control message, adds its

own agenda to it and then forwards it to the next node along the path. When reaching the receiver, the control message will contain all the agendas from ports and network equipments that it has crossed.

Algorithm 1 Scheduling algorithm

```
if the availability agenda of the path is empty then
    Put the reservation in the middle of the remaining period before the deadline, if possible. Otherwise, put it now (+ε for the request processing time).
else
    if there is no event before the deadline then
        Put the reservation in the middle of the remaining period before the deadline if possible. Otherwise, put it as soon as possible.
    else
        foreach event in the availability agenda of the path and while it occurs before deadline do
            Try to place the reservation after and before the event. Memorize the possible places (no collision with other reservations and end-before deadline).
        if there is no possible place then
            if the reservation can be put before the deadline then
                Put the reservation now (+ε for the request's processing time).
            else
                if some events were not possible because of the deadline constraint then
                    if the reservation can be put now (some bandwidth is available) without respecting the deadline then
                        Propose this solution to the user.
                    else
                        foreach of these remaining events while no solution has been found do
                            Try to place the reservation after the event without respecting the deadline. Store the earliest possible place (no collision with other reservations) to propose it to the user.
        else
            foreach possible place do
                Estimate the energy consumption of the transfer using each equipment's energy-cost profiles.
            if there is one less energy-consuming solution then
                Take that place.
            else
                Take the earliest place among the less energy-consuming ones.
```

When the receiver gateway gets the control message, it merges these agendas to obtain an availability agenda per end-to-end path.

C. Scheduling algorithm

The end-to-end availability agenda is scanned using the HERMES scheduling algorithm (Algorithm 1) to find the solution consuming the least amount of energy. At each trial, the solution attempts to use as much bandwidth as possible to reduce the reservation's duration, and hence its cost. We estimate the energy consumption of each possible solution (i.e. place in the agenda) using the Energy Consumption mOdel For End-to-end Networks (ECOFEN) [19], compare the solutions, and pick the least energy consuming. When the receiver gateway schedules the reservation, it sends a *notification message* back along the reservation path to update the agendas and to notify the user of the acceptance or rejection of its request.

The cost is quadratic in the number of considered events. However, that is not an issue since the number of events is limited for three reasons: i) for each reservation, at each time, the algorithm tries to use as much bandwidth as possible, minimizing the number of events added by the reservation; ii) the algorithm only uses the part of the agenda that is concerned by the reservation (between the earliest possible start time and the deadline) and not all the future events, and iii) the algorithm always tries to aggregate reservations, so a new reservation creates less events than if it was not aggregated.

D. Locking mechanism

When scheduling a reservation, it is mandatory to guarantee that requests are not scheduled to use the same time slot/bandwidth in one or more agendas. The scheduling algorithms found in literature does not take care of this issue. We use a locking mechanism to ensure no overlaps of reservations.

Algorithm 2 Management of the locks on a router.

```
foreach lock on an agenda do
    if the lock timer initiated for a time period τ expires then
        Release the lock;
        Inform the gateway concerned by the next lock on this agenda and start this new lock and its timer;
foreach notification message (after the request scheduling stage) do
    foreach concerned agenda do
        update the agenda if necessary;
        stop the concerned timer, the remaining time is denoted t_τ;
        release the lock concerning this request;
        if there is a lock after the released one then
            if t_τ is greater than θ_τ and the agenda has been updated then
                inform the gateway scheduling the request concerned by this lock (update message);
            put this new lock on the concerned part of the agenda and start its timer;
foreach incoming request (control message) do
    foreach concerned agenda do
        if there is no lock on the concerned part of the agenda then
            put a lock on this part of the agenda;
        else
            put a lock on this part of the agenda after the last lock;
            remember the gateway to contact at the release of the lock that precedes this new lock;
```

When the control message collecting the agendas goes through a node, the node is informed of the part of its agendas required by the reservation. Then, the concerned parts of the agendas are locked until the control message returns with updates. For fault-tolerance reasons, a timer is used on each node, in which the *default lock time* is denoted by τ. This timer is activated when an agenda is locked, and when it expires, the lock is released.

Only the ports' agendas are concerned with this locking mechanism. As the routers' agendas are used only during scheduling to find the most energy-efficient reservations, they are not affected by overlapping of concurrent requests. On a router, the overlapping requests are those that are being scheduled at the same time using the same interfaces.

When part of an agenda is locked, no other incoming request can reserve this part. Instead, the incoming request tries to place a new lock to be activated once the current lock is released. When the current lock expires, the node sends an *update message* to the gateway that is scheduling the reservation that required the next lock. This message contains the updates made on the agenda during the previous lock.

In order to minimize the number of messages transferred across the network, if a lock is released less than θ_τ before it expires and there is no update on the agenda, no message is sent to the gateway concerned by the next lock. A message is unnecessary since the agenda has not been updated and the gateway already knows when its lock period begins.

The values of τ and θ_τ are common to the entire network and are based on the average latency between any two nodes. The rationale is that update messages are not required if they will arrive at a gateway after the start of its lock period. Algorithm 2, which runs as an infinite loop, shows how a router deals with agenda locks and timers. It also presents the process used to handle concurrent requests.

Algorithm 3 Modification of the scheduling algorithm to include the locking mechanism.

```
if None of the concerned agendas has a lock then
    Use the normal scheduling algorithm;
else
    Start the scheduling algorithm by estimating the energy consumption at each possible place;
    foreach lock do
        Wait for the update message (sent after the lock release) until the beginning of its lock time slot;
    Estimate the energy consumption at new places created by the updates;
    Delete non-possible places due to the updates;
    Pick the least energy-consuming place;
    Send back the notification message to notify the user, to update the concerned agendas and to release the corresponding locks;
```

The request scheduling process is slightly impacted by this locking mechanism as shown in Algorithm 3, which describes the modified scheduling process at the receiver gateway. This algorithm improves the scheduling algorithm presented in [18] (Algorithm 1) by executing the locking mechanism once the agendas are merged. The algorithms described above respect the principle of fairness amongst requests since they are treated and scheduled in the order of their submission times.

E. Re-planning capacity

We have presented algorithms that work in a static way, which means that once a decision has been taken, it cannot be changed. Our re-planning capacity allows reservations to move dynamically after their registration in the path agendas, still guaranteeing the same QoS and respecting the user deadline. Off-line algorithms can often lead to optimal solutions in terms of energy conservation because all the reservation requests are known from the start. However, it is not our case since we use on-line algorithms. The general idea of these new features has been introduced in [18] as an adaptive functionality.

Algorithm 4 Re-planning of reservations to save more energy.

```
E ← {R_b, R_e};
while M ≠ do
    foreach event t in E which is smaller than M[0] do
        Try to place M[0] before and after t;
    Estimate the energy consumption for each possible place (including the current one);
    Pick the least consuming one;
    if the picked place is different from the current one then
        Store this update;
        Add the beginning and end of this moved reservation in E;
    Remove M[0] from M;
Send all the updates in the notification message;
```

In our distributed case, each node has partial information about the agendas of all the other nodes. Moreover, as agendas are inter-dependent, if you change a reservation in one agenda, all the other agendas, which can be on different nodes, need to change. Hence, off-line algorithms cannot be applied directly in this case. To design the re-planning capacity, we use the path agenda generated by the gateway's merging operation before the scheduling process. When the reservation is scheduled, the gateway has all the reservations that start on the first node of the path and end on the last node (or reverse path). As the gateway has all the agendas, it can move them.

After scheduling a reservation R, if there are other reservations using the exact same path to start after R begins, they could be re-scheduled and aggregated with R if doing so saves more energy than the current schedule. The re-planning algorithm is detailed in Algorithm 4, where E is the list of

new events, R_b is the start date of reservation R, R_e is the end date of reservation R, and M is the set of reservations occurring after the start of R on the exact same path.

IV. Evaluation

A. BoNeS: Bookable Network Simulator

To validate our model, we use Bookable Network Simulator (BoNeS) [17], a network simulator written in Python. BoNeS takes as input a network-description file (topology and router and link capacities) and some characteristics of the network traffic (e.g., statistical distribution of inter-arrival time of submissions, distribution of the reservation durations, source and destination nodes, distribution of the deadlines and TTLs). Then, it generates the network topology and traffic according to these input characteristics, and after that, simulates different scheduling algorithms, comparing them in terms of both performance and energy consumption.

The simulator runs 5 different scheduling techniques on the generated traffic and network: 1) *first*: the reservation is scheduled at the earliest possible place in the merged agenda; 2) *first green*: the reservation is aggregated with the first possible reservation already accepted (before deadline), or scheduled at the earliest possible place; 3) *last*: the reservation is scheduled at the latest possible place (before deadline); 4) *last green*: the reservation is aggregated with the latest possible reservation already accepted (before deadline); 5) *green*: the implementation of our model, wherein the energy consumption is estimated for each possible allocation, and the least consuming is chosen.

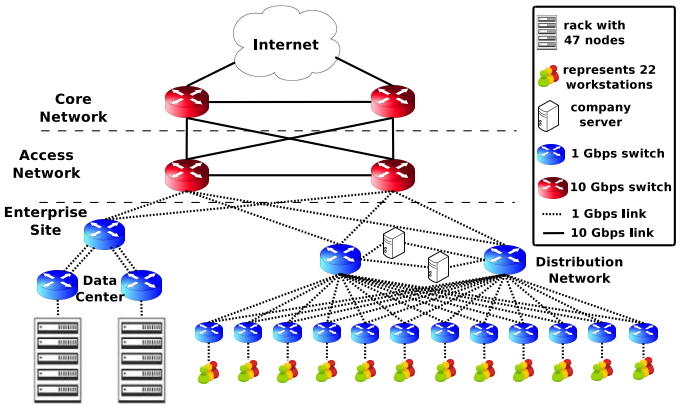

Fig. 1. Example of an enterprise network's topology.

Our simulator evaluates the energy consumption of these five scheduling options combined with the proposed on/off algorithm where unused resources are switched off. For the sake of comparison, the simulator also computes the energy consumption of the *first* schedule without on/off or rate adaptation algorithm (the present scenario), this case is called *no off*. The generated network traffic consists of requests with:

- submission times distributed according to a log-normal distribution;
- data volumes generated with a negative exponential distribution;
- sources and destinations chosen randomly (equiprobability) among the end-hosts; and
- deadlines generated according to a Poisson distribution.

The distributions used in the experiments reported next, have been inspired by the results presented in [7], [3].

As we focus on networking equipment, we disregard the overall energy consumed by servers, taking into account only the consumption of their Ethernet cards. The power consumption of each network equipment is computed using the ECOFEN model [19] combined with realistic values obtained from the literature. Time to boot and to shut down ports and routers are also taken into consideration.

B. Results

We have firstly simulated the core of the Fedwire Interbank Payment Network presented in [22], which represents 75% of its daily amount of transferred data. This network comprises 66 nodes and 181 links, where 25 nodes form a densely connected network to which the remaining nodes connect. Our results for the entire network are provided in Table I. Each node is considered to consume 4 kW at full load [1]. For each value, 80 experiments of 4 hours of simulated time were performed with randomly generated requests simulating 20% of load on each node. One can observe that our model (*green*) provides the best results in terms of cost (in Wh per Tb sent in this network for this traffic) which is a trade-off between energy consumption and number of accepted requests. Compared to the actual management (*no off*), it saves 42% of the energy actually consumed. As we can see, the most relevant value is the cost, since the consumption depends also on the accepted request durations.

Secondly, Figure 1 presents the enterprise network we simulate. Its hierarchical topology, inspired from [16], comprises 21 routers, 360 end-hosts and 404 links. The data center hosts computing resources, databases and storage systems. The two company servers represent the machines hosting typical enterprise services (i.e. DNS, DHCP or LDAP servers). The routers can have advanced functionalities such as firewalls and packet filters.

Table II presents the results for the enterprise network with average load of 25% on the links. Our model still provides the best results for the cost and could save 22% of the energy when using equipments that are able to perform rate adaptation and to switch on and off ports and entire routers.

V. Conclusion and Future Work

Rising bandwidth demands lead to an urgent need for energy-efficient network equipments. However, green solutions should be tailored to each type of network (traffic, topology, usage, etc.) in order to be really efficient. We decided to focus on private networks since they are under the control of a single administrative domain that allows for network-wide management.

This paper presented new features that complete our proposed model termed as HERMES. HERMES is a network overlay that uses bandwidth reservations and combines on/off management, energy-efficient scheduling, reservation aggregation,

Scheduling	First	First green	Last	Last green	Green	No off
Average (Wh)	606 542	604 531	606 819	605 199	605 743	1 050 544
Standard deviation	7 106	6 994	7 012	7 010	7 090	1 511
Accepted requests	98.6%	98.3%	97.9%	97.5%	98.6%	98.6%
Cost in Wh per Tb	4 113	4 119	4 163	4 176	4 106	7 123

TABLE I
ENERGY CONSUMPTION FOR THE CORE OF THE FEDWIRE INTERBANK PAYMENT NETWORK.

Scheduling	First	First green	Last	Last green	Green	No off
Average (Wh)	80 987	80 990	80 255	78 611	80 335	103 466
Standard deviation	392	405	316	333	307	563
Accepted volume (Tb)	19.402	19.306	19.098	17.748	19.400	19.402
Cost in Wh per Tb	4 174	4 195	4 202	4 429	4 141	5 333

TABLE II
ENERGY CONSUMPTION FOR THE ENTERPRISE NETWORK.

and usage prediction to find a trade-off between conserving energy and meeting performance requirements. The new features include an analysis of the cost of the scheduling algorithm, a locking mechanism, a re-planning capacity technique and the architecture of routers and gateways.

A new evaluation of HERMES on realistic scenarios is also presented, conducted with our simulator, BoNeS, using real power profiles for network equipments. Our results show that, under the considered traffic and topology scenarios, a significant amount of energy could be saved compared to the present consumption.

Our future work comprises adapting HERMES to all types of networks to also deal with small flows (mice), since HERMES is currently more suitable for bulk transfers or big flows (elephants). The idea is to aggregate small flows to obtain medium-sized flows (dogs). We also plan to deal with urgent flows either by provisioning bandwidth in advance using prediction algorithms or by always letting a low-power spanning tree network (sub-network of the initial network) powered on to transmit urgent messages. The problem of transmitting urgent messages occurs only when parts of the network are in sleep-mode. We also intend to make our simulator BoNeS publicly available.

ACKNOWLEDGMENT

We thank Marcos Dias de Assunção (INRIA, France) for his feedback and useful comments on this work.

REFERENCES

[1] J. Baliga, R. Ayre, K. Hinton, and R. Tucker, "Green Cloud Computing: Balancing Energy in Processing, Storage, and Transport," *Proceedings of the IEEE*, vol. 99, no. 1, pp. 149–167, 2011.

[2] C. Becher, S. Millard, and K. Soramaki, "The network topology of CHAPS Sterling," Bank of England working papers (No. 355), 2008.

[3] T. Benson, A. Akella, and D. Maltz, "Network traffic characteristics of data centers in the wild," in *Conference on Internet measurement (IMC)*, 2010, pp. 267–280.

[4] M. Bossa, H. Elsingerb, M. Summera, and S. Thurner, "Network topology of the interbank market," *Quantitative Finance*, vol. 4, no. 8, pp. 677–684, 2004.

[5] L.-O. Burchard, "Networks with Advance Reservations: Applications, Architecture, and Performance," *Journal of Network and Systems Management*, vol. 13, no. 4, pp. 429–449, 2005.

[6] K. Christensen, C. Gunaratne, B. Nordman, and A. George, "The next frontier for communications networks: power management," *Computer Communications*, vol. 27, no. 18, pp. 1758–1770, 2004.

[7] D. Ersoz, M. Yousif, and C. Das, "Characterizing Network Traffic in a Cluster-based, Multi-tier Data Center," in *International Conference on Distributed Computing Systems (ICDCS)*, 2007.

[8] C. Gunaratne, K. Christensen, and B. Nordman, "Managing Energy Consumption Costs In Desktop PCs And LAN Switches With Proxying, Split TCP Connections, And Scaling Of Link Speed," *International Journal Of Network Management*, vol. 15, no. 5, pp. 297–310, 2005.

[9] M. Gupta and S. Singh, "Greening Of The Internet," in *SIGCOMM '03*, 2003, pp. 19–26.

[10] ——, "Dynamic Ethernet Link Shutdown For Energy Conservation On Ethernet Links," in *International Conference On Communications (ICC '07)*, 2007, pp. 6156–6161.

[11] J. Kurian and K. Sarac, "A survey on the design, applications, and enhancements of application-layer overlay networks," *ACM Computing Surveys*, vol. 43, pp. 5:1–5:34, 2010.

[12] Y. Li, S. Ranka, and S. Sahni, "In-advance path reservation for file transfers In e-Science applications," in *IEEE Symposium on Computers and Communications (ISCC 2009)*, 2009, pp. 176–181.

[13] P. Mahadevan, P. Sharma, S. Banerjee, and P. Ranganathan, "Energy Aware Network Operations," in *INFOCOM Workshops*, 2009.

[14] B. Nordman and K. Christensen, "Proxying: The Next Step In Reducing IT Energy Use," *Computer*, vol. 43, no. 1, pp. 91–93, 2010.

[15] A. Odlyzko, "Data Networks are Lightly Utilized, and will Stay that Way," *Review of Network Economics*, vol. 2, 2003.

[16] P. Oppenheimer, *Top-Down Network Design*. Cisco Press, 2010.

[17] A.-C. Orgerie and L. Lefèvre, "Energy-Efficient Framework for Networks of Large-Scale Distributed Systems," in *IEEE International Symposium on Parallel and Distributed Processing with Applications (ISPA)*, 2011.

[18] A.-C. Orgerie, L. Lefèvre, and I. Guérin-Lassous, "Energy-Efficient Bandwidth Reservation for Bulk Data Transfers in Dedicated Wired Networks," *The Journal of SuperComputing*, 2011.

[19] A.-C. Orgerie, L. Lefèvre, I. Guérin-Lassous, and D. Lopez Pacheco, "ECOFEN: an End-to-end energy Cost mOdel and simulator For Evaluating power consumption in large-scale Networks," in *International Workshop on Sustainable Internet and Internet for Sustainability (Sustainet)*, 2011.

[20] R. Pang, M. Allman, M. Bennett, J. Lee, V. Paxson, and B. Tierney, "A first look at modern enterprise traffic," in *Conference on Internet Measurement (IMC)*, 2005.

[21] W. Reinhardt, "Advance Reservation of Network Resources for Multimedia Applications," in *International Workshop on Multimedia (IWAKA)*, 1994, pp. 23–33.

[22] K. Soramki, M. Bech, J. Arnold, R. Glass, and W. Beyeler, "The Topology of Interbank Payment Flows," *Physica A: Statistical Mechanics and Its Applications*, vol. 379, no. 1, pp. 317–333, 2007.