Green Networking With Packet Processing Engines: Modeling and Optimization

Raffaele Bolla, *Member, IEEE*, Roberto Bruschi, *Member, IEEE*, Alessandro Carrega, *Student Member, IEEE*, and Franco Davoli, *Senior Member, IEEE*

***Abstract*—With the aim of controlling power consumption in metro/transport and core networks, we consider energy-aware devices able to reduce their energy requirements by adapting their performance. In particular, we focus on state-of-the-art packet processing engines, which generally represent the most energy-consuming components of network devices, and which are often composed of a number of parallel pipelines to "divide and conquer" the incoming traffic load. Our goal is to control both the power configuration of pipelines and the way to distribute traffic flows among them. We propose an analytical model to accurately represent the impact of green network technologies (i.e., low power idle and adaptive rate) on network- and energy-aware performance indexes. The model has been validated with experimental results, performed by using energy-aware software routers loaded by real-world traffic traces. The achieved results demonstrate how the proposed model can effectively represent energy- and network-aware performance indexes. On this basis, we propose a constrained optimization policy, which seeks the best tradeoff between power consumption and packet latency times. The procedure aims at dynamically adapting the energy-aware device configuration to minimize energy consumption while coping with incoming traffic volumes and meeting network performance constraints. In order to deeply understand the impact of such policy, a number of tests have been performed by using experimental data from software router architectures and real-world traffic traces.**

***Index Terms*—Adaptive rate, forwarding engine, green networking, low power idle, multipipeline.**

I. Introduction

In THE last few years, power consumption has shown a growing and alarming trend in all industrial sectors, and particularly in information and communication technology (ICT) [1]. Public organizations, Internet service providers (ISPs), and telecom operators reported alarming statistics of network energy requirements and of the related carbon footprint [2].

The Global e-Sustainability Initiative (GeSI) estimated the overall carbon footprint of European network devices and infrastructure to be about 349 $MtCO_2e$ (Mt equivalent of CO_2) in 2020, with a 131% increase with respect to 2007 if no green network technologies (GNTs) would be adopted [3].

The study of power-saving network devices has been based in recent years on the possibility of adapting network energy requirements to the actual traffic load. Indeed, it is well known that network links and devices are generally provisioned for busy or rush-hour load, which typically exceeds their average utilization by a wide margin [4]. Although this margin is seldom reached, network devices are designed on its basis and, consequently, their power consumption remains more or less constant even in the presence of fluctuating traffic load. Thus, the key of any advanced power saving criteria resides in dynamically adapting resources, provided at the network, link, or equipment level, to current traffic requirements and loads [5]–[7]. In this respect, current green networking approaches have been based on numerous energy-related criteria, to be applied in particular to network equipment and component interfaces [6], [7].

Despite the great progresses of optics in transmission and switching, it is well known that today's networks still rely very strongly on electronics. Indeed, the operational power requirements arise from all the hardware (HW) elements realizing network-specific functionalities, like the ones related to data- and control-planes, as well as from elements devoted to auxiliary functionalities (e.g., air cooling, power supply, etc.). In this respect, the data plane certainly represents the most energy-consuming and critical element in the largest part of network device architectures since it is generally composed by special-purpose HW elements (packet processing engines, network interfaces, etc.) that have to perform per-packet forwarding operations at very high speeds.

Tucker *et al.* [8] and Neilson [9] focused on high-end IP routers and estimated that the power required at the data plane weighs for 54% on the overall device architectures, versus 11% for the control plane and 35% for power and heat management. The same authors further broke out energy consumption sources at the data plane on a per-functionality basis. Internal packet processing engines require about 60% of the power consumption at the data plane of a high-end router, network interfaces weigh for 13%, switching fabric for 18.5%, and buffer management for 8.5%.

Manuscript received November 30, 2011; revised November 12, 2012; accepted January 17, 2013; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor P. Crowley. Date of publication February 14, 2013; date of current version February 12, 2014. This work was supported by the European Commission under the 7th Framework Programme (FP7) ECONET (low Energy Consumption NETworks) project, the Italian Ministry of Education, University and Research (MIUR) under the EFFICIENT (Energy eFFIcient teChnologIEs for the Networks of Tomorrow) PRIN project, and the GreenNet (Greening the Network) FIRB project.

R. Bolla, A. Carrega, and F. Davoli are with the Department of Electrical, Electronic and Telecommunication Engineering, and Naval Architecture (DITEN), University of Genoa, and the National Inter-University Consortium for Telecommunications (CNIT), University of Genoa Research Unit, 16145 Genova, Italy (e-mail: raffaele.bolla@unige.it; alessandro.carrega@unige.it; franco.davoli@unige.it).

R. Bruschi is with the National Inter-University Consortium for Telecommunications (CNIT), University of Genoa Research Unit, 16145 Genova, Italy (e-mail: roberto.bruschi@cnit.it).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNET.2013.2242485

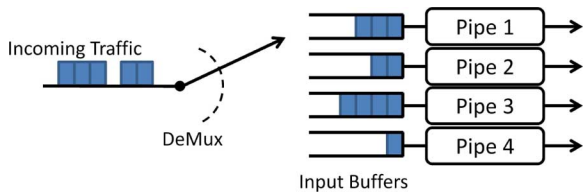

Fig. 1. Scheme of the considered architecture: The traffic incoming from a SerDes bus is demultiplexed by a load-balancer component toward multiple parallel pipelines in a packet processing engine.

Starting from these data, we decided to focus on packet processing engines, which generally represent the most energy-hungry physical components, even for other network devices, besides high-end routers. These engines are realized with heterogeneous HW technologies [from classical application-specific integrated circuit (ASIC) [10] or field programmable gate array (FPGA) [11] chips to the ones based on graphical processing units (GPUs) [12]] and often have highly parallel architectures in order "to divide and conquer" the traffic load incoming from a number of high-speed interfaces.

We consider the architecture shown in Fig. 1. The traffic flows enter and exit the engine by means of Serializer/Deserializer busses (SerDes), which may be realized with different standards, like Peripheral Component Interconnect (PCI) Express, Serial Gigabit Media Independent Interface (SGMII), 10 Gigabit Media Independent Interface (XGMII), Attachment Unit Interface (XAUI), etc. In high-performance architectures, a specific HW component is required in order to multiplex and demultiplex traffic between the SerDes and the parallel pipelines of the engine. This component can be included inside the same packet processing engine [10], or it can be placed in the interface cards before the SerDes bus [like in the Receive-Side Scaling (RSS) standard for server network interface cards [13]]. In addition, it is worth noting that we focus on packet processing engines with multiple parallel pipelines inside a single line-card. In this case, we can assume that each pipeline can adapt its rate separately, and that all incoming flows can be handled by any subset of the pipelines. This possibility would not extend to the case of scheduling traffic among multiple line-cards.

In this architecture, the workload distribution can be critical due to the presence of multiple parallel pipelines. Several studies have been made on how to distribute the load among different resources and propose different algorithms [14]–[17]. In [17], the authors focus on how to preserve the packet ordering within individual TCP connections and to achieve both load balancing and efficient system utilization. Reference [14] examines the sources of load imbalances in hash-based scheduling schemes and applies different scheduling policies. However, the problem of packet ordering can be avoided by considering that modern network processing units (NPUs) include dedicated HW for packet reordering. For example, the Netlogic XLP NPU provides the Packet Ordering Engine (POE) [10]. In this respect, we do not consider a specifc scheduling/reordering algorithm to be used along with the load distribution procedure among the pipelines.

In such scenario, we assume to adopt two basic techniques, already heavily widespread in silicon technologies, in order to reduce the energy requirements of the packet processing engine: adaptive rate (AR) and low power idle (LPI) [6]. The former allows dynamically modulating the capacity of a processing engine (or of a single pipeline) in order to meet traffic loads and service requirements, while the latter forces processing engines (or single pipelines) to enter low-power states when not sending/processing packets. As outlined in a number of previous works [4], [6], [18], the use of such techniques generally allows trading energy consumption for network performance (in terms of packet latency times, loss rate, etc.).[1]

Assuming the possibility of selectively tuning AR and LPI mechanisms for each parallel pipeline, our goal is to dynamically manage the engine configuration in order to optimally balance its energy consumption with respect to its network performance. Given the incoming load features and parameters, we want to find: 1) how many pipelines have to actively work; 2) their AR and LPI configurations; and 3) which share of the incoming traffic volume the load balancer module must assign to them.

To this purpose, our main objective is to provide a novel analytical model based on classical concepts of queuing theory and able to capture the tradeoff between energy- and network-aware performance metrics when AR and/or LPI techniques are adopted in a network device. We model the energy- and network-aware dynamics of packet processing engines and formalize an optimization problem in a fairly general way to reflect different criteria like the following:

1) the minimization of energy consumption under certain constraints on packet latency times and packet loss rate;
2) the maximization of network performance for a given energy cap;
3) the optimization of a given tradeoff between the two previous objectives.

According to a given criterion, the optimization problems we consider take explicitly into account maximum allowed energy consumption or packet latency and loss constraints.

This paper is organized as follows. Section II introduces AR and LPI capabilities and how they can impact on network performance. The proposed model is described in Section III, and its validation results are in Section IV. The optimization procedure based on the proposed model for the energy-aware load balancing with multiple pipelines is explained in Section V. Section V-B illustrates the tradeoff flexibility for the multiple pipeline case, and Section VI shows some performance evaluation results obtained with the optimization procedure. Finally, the conclusions are drawn in Section VII.

II. Energy-Aware Silicon and Network Performance

The current generation of network devices does not support power-scaling functionalities. However, power management is a key feature in today's processors across all market segments, and it is rapidly evolving also in other HW technologies [19]. In this section, we first introduce the Advanced Configuration and Power Interface (ACPI) specification [20] and how it makes AR and LPI capabilities accessible at the software (SW) layer. Then, in the second part, we discuss the impact of AR and LPI on the

[1]For the sake of clarity, even though LPI might be seen as a limiting case of AR, we prefer to explicitly distinguish the two techniques.

forwarding performance of a network device and how these two capabilities may interact between themselves.

A. ACPI Specification

In general-purpose computing systems, the ACPI specification provides an open industrial standard for device configuration and power management by the operating system. This standard models the AR and LPI functionalities by introducing two sets of energy-aware states, namely, performance and power states (P- and C-states), respectively.

Regarding the C-states, C_0 indicates the operating state where the central processing unit (CPU) executes instructions, while C_1–C_X are processor LPI states. As the sleeping power state $(C_1, \ldots, C_X)$ becomes deeper, less power is consumed, but the transition between active and sleeping (and vice versa) requires longer time. In particular, C_1 is a state where the processor is not executing instructions, but can return to the C_0 state essentially instantaneously. All ACPI-conformant processors must support this power state. Excluding C_0, the number of LPI states is optional, and the relative characteristics (transition times, power saving compared to C_0 power consumption, etc.) depend on the specific platform implementation.

Instead, the performance of the processor's cores is tuned through P-state transitions. P-states allow modifying the operating energy point of a processor/core by altering the working frequency and/or voltage. Thus, by using P-states, a core can consume different amounts of power while providing different processing performance at the C_0 state. At a given P-state, the core can transit to higher C-states in idle conditions. In general, the higher the index of P- and C-states is, the less will be the power consumed and the heat dissipated. These states are implementation-dependent, but P_0 is always the highest performance state, with P_1–P_Y being successively lower-performance states, up to an implementation-specific limit of Y not greater than 16.

Due to issues in silicon electrical stability, the transition between different P-states is generally very slow with respect to packet processing times: A large part of current CPUs can switch their operating P-state in about 10 ms.[2] Given such large P-state transition times, it is worth noting that any AR-based closed-loop control policies with tight time constraints may be generally not feasible and might not be adopted for optimizing power consumption inside network device architectures.

B. Energy-Aware Tradeoffs

As previously sketched, LPI and AR have different impacts on packet forwarding performance. Fig. 2 shows how AR [Fig. 2(c)] causes a stretching of packet service times, while the sole adoption of LPI [Fig. 2(b)] introduces an additional delay in packet service, due to the wake-up times. Moreover, preliminary studies in this field [4] showed how performance scaling and idle logic work like traffic-shaping mechanisms by causing opposite effects on the traffic burstiness level. The wake-up times in LPI favor packet grouping and then an increase in traffic burstiness, while service time expansion in AR favors burst untying and consequently traffic profile smoothing. Finally, as outlined in Fig. 2(d), the joint adoption of both energy-aware capabilities may not necessarily lead to outstanding energy gains since performance scaling causes larger packet service times and consequently shorter idle periods. It is worth noting that the overall energy saving and the network performance strictly depend on incoming traffic volumes and statistical features (interarrival times, burstiness levels, etc.). For instance, idle logic provides top energy and network performance when the incoming traffic has a high burstiness level. This is because less active–idle transitions (and wake-up times) are needed, and the HW can remain in a low consumption state for longer periods.

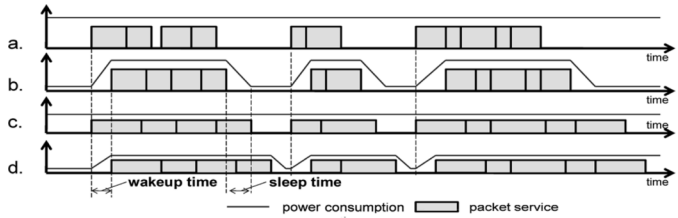

Fig. 2. Packet service times and power consumptions in the cases with: (a) no power-aware optimizations, (b) only LPI, (c) only AR, and (d) AR and LPI.

III. Analytical Model

In this section, we introduce a model already proposed and analyzed in our previous work [18] in the single pipeline case, which aims at representing the behavior and the performance of an energy-aware network device that includes LPI and AR capabilities. For the sake of simplicity, we adopt the ACPI representation of power management primitives and refer to AR and LPI configurations in terms of P- and C-states.

We consider a traffic demultiplexer distributing the incoming traffic among Λ parallel pipelines. For the ith pipeline, we assume to model the packet forwarding engine as a single-server queuing system with maximum service rate $\mu^{(i)}$, $i \in \{1, \ldots, \Lambda\}$. Similarly to [21] and [22], the service rate $\mu^{(i)}$ represents the device capacity in terms of packet headers that can be processed per second. Moreover, we assume all packet headers requiring a constant service time. This hypothesis represents a reasonable approximation for a large part of current routing and switching devices. A finite buffer, with size equal to $N^{(i)}$ packets, is assumed to be bound to the server for backlogging incoming traffic of the ith pipeline. The model notation is introduced in Table I. The selection of different P- and C-states is supposed to impact on the forwarding engine performance in terms of both packet service capacity and wake-up times of the servers.

In order to make the equations in the rest of this section more readable, we omit the index $^{(i)}$ of the pipeline. Thus, we refer to the arrival rate $\lambda^{(i)}$ of the ith pipeline or to the device capacity $\mu^{(i)}$ with λ and μ, respectively, and we do so for all other quantities.

The rest of this section is organized as follows. Section III-A introduces the main parameters to be considered in a device with

[2]Actually, CPU switching time can be in the order of 10 μs or less; substantial additional latency can be added by the operating system. Fast network processors can switch P-states in 10–100 μs.

TABLE I
NOTATION

Symbol	Description
Λ	number of parallel pipelines.
$C_x^{(i)}$	selected $C-$state of the i^{th} pipeline, $C_x^{(i)} \in \left\{C_1^{(i)}, \ldots, C_X^{(i)}\right\}$.
$P_y^{(i)}$	selected $P-$state of the i^{th} pipeline, $P_y^{(i)} \in \left\{P_0^{(i)}, \ldots, P_Y^{(i)}\right\}$.
$\tau_{on}^{(i)} = \tau_{on}\left(C_x^{(i)}\right)$	time needed to wake up the HW of the i^{th} pipeline from the $C_x^{(i)}$ sleeping state.
$\tau_{off}^{(i)} = \tau_{off}\left(C_x^{(i)}\right)$	time needed to put the active HW of the i^{th} pipeline into the $C_x^{(i)}$ sleeping state.
$\tau_r^{(i)} = \tau_r\left(P_y^{(i)}\right)$	time to recover forwarding operation (in state $P_y^{(i)}$) after the HW wake-up of the i^{th} pipeline.
$\tau_s^{(i)} = \tau_s\left(P_y^{(i)}, C_x^{(i)}\right)$ $= \tau_{on}^{(i)} + \tau_r^{(i)}$	setup time of the i^{th} pipeline in the transition from $C_x^{(i)}$ to $P_y^{(i)}$.
$\mu^{(i)} = \mu\left(P_y^{(i)}\right)$	packet service rate of the i^{th} pipeline in the $P_y^{(i)}$ state.
$\Phi_a^{(i)} = \Phi_a\left(P_y^{(i)}\right)$	power consumption when the i^{th} pipeline is active in $P_y^{(i)}$ state.
$\Phi_{idle}^{(i)} = \Phi_{idle}\left(C_x^{(i)}\right)$	power consumption when the i^{th} pipeline is sleeping in $C_x^{(i)}$ state.
$\Phi_t^{(i)} = \Phi_t\left(C_x^{(i)}\right)$	power consumption during $\tau_{on}\left(C_x^{(i)}\right)$ and $\tau_{off}\left(C_x^{(i)}\right)$ periods.
$N^{(i)}$	buffer size of the i^{th} pipeline.
$\lambda^{(i)}$	rate of batch arrival to the i^{th} pipeline.
$\hat{\lambda} = \sum_{l=1}^{\Lambda} \lambda^{(l)}$	rate of batch arrival to the demultiplexer.
$\hat{\beta}_j$	probability that an incoming burst to the demultiplexer contains j packets.
$\beta_j^{(i)} = \hat{\beta}_j$ (if SBSP)	probability that an incoming burst to the i^{th} pipeline contains j packets.
$\hat{X}(z)$	Probability Generating Function (PGF) of batch sizes incoming to the demultiplexer.
$X^{(i)}(z) = \hat{X}(z)$ (if SBSP)	Probability Generating Function (PGF) of batch sizes incoming to the i^{th} pipeline.
$\hat{\beta}$	average number of customers in an incoming batch to the demultiplexer.
$\beta^{(i)} = \hat{\beta}$ (if SBSP)	average number of customers in an incoming batch to the i^{th} pipeline.
$P_k^{(i)}$	stationary probability of having $k \in \left[0, N^{(i)}\right]$ packets in the i^{th} pipeline.
$\rho^{(i)} = \frac{\lambda^{(i)}\beta^{(i)}}{\mu^{(i)}}$ (with $N^{(i)} \to \infty$)	utilization factor of the i^{th} pipeline.
$T_I^{(i,n)}$	average duration of the i^{th} pipeline idle periods including $\tau_{off}^{(i)}$.
$T_B^{(i,n)}$	average duration of the i^{th} pipeline busy periods including $\tau_s^{(i)}$.
$T_R^{(i,n)} = T_I^{(i,n)} + T_B^{(i,n)}$	average duration of idle/busy renewal cycle.
$B^{(i)}(s)$	Laplace transform of the i^{th} pipeline customer service process.
$V^{(i)}(s)$	Laplace transform of the i^{th} pipeline vacation process due to setup times.
$P_{loss}^{(i)}$	packet loss probability of the i^{th} pipeline.

AR and LPI capabilities. Section III-B shows the model representing the traffic incoming to the energy-aware device. Finally, the proposed analytical model of the pipeline is described in Sections III-C–III-F.

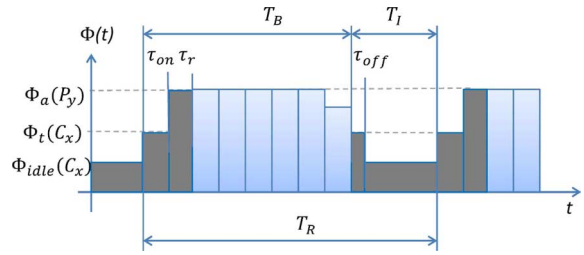

Fig. 3. Power consumptions during a renewal busy–idle cycle.

A. Introducing the Energy-Aware Parameters

In this section, we consider the notation for a generic pipeline. Let $\{C_1, \ldots, C_X\}$ and $\{P_0, \ldots, P_Y\}$ be the sets of sleeping and performance states, respectively, available in the pipeline. State C_0 is the active state, where the processor is working in a given P_y configuration.

Each sleeping state is thought to be bound with both a different value of idle power consumption Φ_{idle} and different transition times τ_{off} and τ_{on} needed to enter and to wake up from the idle state, respectively. In a similar way, each P-state can be related with active power consumption Φ_{a}, as well as with a packet processing capacity μ. As the P_y state becomes higher, both Φ_{a} and μ values decrease.

From the considerations in Section II, it is reasonable to assume the network device working at small timescales by switching between a sleeping C_x state when idle and a running P_y state when performing operations. For this reason, throughout the paper we do not explicitly indicate the dependency of parameters on the P- and C-states, which are shown in Table I.

For each fixed state pair $\{C_x, P_y\}$, the system works with the renewal process representation shown in Fig. 3. The server representing the pipeline has infinitely many alternating busy $T_{\text{B}}{}^{(n)}$ and idle $T_{\text{I}}{}^{(n)}$ periods, where the index n denotes the order of the interval. During a generic $T_{\text{B}}{}^{(n)}$, the server is active and performing packet-forwarding activities, and then has instantaneous power consumption equal to Φ_{a}. Afterwards, when it serves the last backlogged packet, it enters the $T_{\text{I}}{}^{(n)}$ period corresponding to the low-consumption C_x state.

As mentioned, transitions from the active state C_0 to a C_x state are not instantaneous, and a transition time τ_{off} is required. When new packets are received, the device has to wake up by exiting the C_x state and returning to the active one (this requires an additional τ_{on} period). Furthermore, depending on the specific device architecture and implementation, an additional time τ_{r} is required to setup and to suitably reconfigure the packet elaboration process. It is worth noting that while τ_{on} and τ_{off} depend on the sleeping C_x state, the τ_{r} parameter depends on the P_y state since it represents a certain number of operations that have to be performed by the server before starting packet forwarding. The instantaneous power requirements of a generic pipeline (Pl) can be expressed as follows:

$$\Phi = \begin{cases} \Phi_{\text{idle}}(C_x), & \text{if Pl is in } C_x \\ \Phi_{\text{a}}(P_y), & \text{if Pl is in } P_y \\ \Phi_{\text{t}}(C_x), & \text{if Pl is in } C_0 \to C_x \text{ or } C_x \to C_0. \end{cases} \quad (1)$$

As in most commercial off-the-shelf (COTS) platforms, $\tau_{\text{off}} \ll \tau_{\text{on}}$, in the model derived in this paper, we neglect the τ_{off} period.

B. Traffic Model

The modeling and the statistical characterization of packet interarrival times are well known to have long-range dependency (LRD) and multifractal statistical features [23], [24]. However, as sustained more recently in [25] and [26], a batch Markov arrival process (BMAP) can effectively estimate the network traffic behavior. Therefore, we decided to model incoming traffic through a BMAP with LRD batch sizes.

We assume that the traffic incoming to the demultiplexer is represented as a BMAP with batch arrival rate $\hat{\lambda}$. In more detail, the demultiplexer receives groups of j packets at exponentially distributed interarrival times with average value equal to $1/\hat{\lambda}$. The sizes j of packet batches are supposed to follow Zipf's law (which can be regarded as the discrete version of a truncated continuous Pareto probability distribution with a certain parameter $\nu > 0$). Therefore, we assume that incoming packet batches have the following probability mass function:

$$\hat{\beta}_j = \begin{cases} \dfrac{1}{j^{\nu} \sum\limits_{l=1}^{j_{\max}} \frac{1}{l^{\nu}}}, & j \le j_{\max} \\ 0, & j > j_{\max} \end{cases} \tag{2}$$

where $\hat{\beta}_j$ represents the probability that an incoming burst contains j packets, with $j \in [1, j_{\max}]$. The average packet number in a batch, $\hat{\beta}$, is then obtained as

$$\hat{\beta} = \frac{\sum\limits_{l=1}^{j_{\max}} \frac{1}{l^{\nu-1}}}{\sum\limits_{l=1}^{j_{\max}} \frac{1}{l^{\nu}}}. \tag{3}$$

Thus, we obtain the probability generating function (PGF) of batch sizes as

$$\hat{X}(z) = \sum_{j=1}^{\infty} \hat{\beta}_j z^j = \sum_{j=1}^{j_{\max}} \frac{z^j}{j^{\nu} \left(\sum\limits_{l=1}^{j_{\max}} \frac{1}{l^{\nu}} \right)}. \tag{4}$$

Starting from the main achievements of previous work [4], and in order to make an optimal use of LPI primitives, we decided not to untie the incoming packet batches, and to send every packet composing a batch to a single pipeline. We call this policy Same Batch Same Pipeline (SBSP); this allows reducing the power consumption of the system at the price of a slight increase in packet latency times, especially at low incoming traffic loads.[3]

Under such assumptions, we can simply deduce that the process of incoming traffic to the Λ pipelines is still BMAP, with the following parameters (we resume the index $^{(i)}$ here for the sake of clarity):

$$\begin{cases} \beta_j^{(i)} = \hat{\beta}_j \\ \beta^{(i)} = \hat{\beta} \\ \hat{\lambda} = \sum_{l=1}^{\Lambda} \lambda^{(l)} \\ X^{(i)}(z) = \hat{X}(z) \end{cases} \quad \begin{array}{l} i = 1, \ldots, \Lambda \\ \text{with SBSP policy.} \end{array} \tag{5}$$

C. Proposed Queuing Model

Considering the assumption described in Section III-B to send every packet composing a batch to a single pipeline, we can outline that the model we propose corresponds to a $M^x/D/1/\text{SET}$ queuing system [27]. For each pipeline, packets arrive in batches with exponentially distributed interarrival times with average rate λ and are served at a fixed rate μ. In order to take the LPI transition periods into account, the model considers deterministic server setup times. When the system becomes empty, the server is turned off. The system returns operational only when a batch of packets arrives. At this point in time, service can begin only after an interval $\tau_{\text{s}} = \tau_{\text{on}} + \tau_{\text{r}}$ has elapsed.

Sections III-D–III-F introduce the analytical model and its specialization to our case. In Section III-D, we derive the PGF and the stationary probabilities of the $M^x/D/1/\text{SET}$ queuing system; in Section III-E we express the server's idle and busy periods. Then, we propose an approximation for the packet loss probability in the case of a finite buffer of size N, and we derive network- and energy-aware performance indexes in Section III-F.

D. PGF and the Stationary Probabilities

In order to obtain the values of stationary probabilities P_k for $n \in [0, \infty]$, we exploit the PGF of the $M^x/G/1$ system as shown in [27] and [28]

$$P(z; M^x/G/1) = (1-\rho) \cdot \frac{(1-z)B(\lambda - \lambda X(z))}{B(\lambda - \lambda X(z)) - z}. \tag{6}$$

Under the assumption that service times are deterministic, we can express the Laplace transform of service times as

$$B(s) = e^{-\frac{s}{\mu}}. \tag{7}$$

Thus, we obtain that the PGF of the $M^x/D/1$ queuing system can be written as

$$P(z; M^x/D/1) = (1-\rho)(1-z) \cdot \frac{e^{-\frac{\lambda}{\mu}[1-X(z)]}}{e^{-\frac{\lambda}{\mu}[1-X(z)]} - z}. \tag{8}$$

By exploiting the stochastic decomposition results of Doshi [29] for the single-unit arrival case and the results in [27] for bulk arrivals, the PGF of the $M^x/G/1$ queue with setup times turns out to be

$$P_{\text{s}}(z) = \zeta(z) P(z; M^x/D/1) \tag{9}$$

[3]The model can be simply and suitably extended to consider untying packet batches also.

where the subscript s indicates the PGF with setup times and

$$\zeta(z) = \frac{1 - zV(\lambda - \lambda X(z))}{\left(\frac{1}{\beta} + \lambda\tau_{\rm s}\right)[1 - X(z)]} \tag{10}$$

is the PGF of the number of arrivals during the residual life of the vacation period, defined as an idle period plus a setup period $\tau_{\rm s}$. Since server setup times have constant durations equal to $\tau_{\rm s}$, we can express the Laplace transform of the vacation process $V(s)$ as

$$V(s) = e^{-\tau_{\rm s} s}. \tag{11}$$

By using (10) and (11) in (9), we can obtain the PGF of our $M^x/D/1/{\rm SET}$ system

$$\begin{aligned} P_{\rm s}(z) = (1-\rho)&\frac{1 - z \cdot e^{-\lambda\tau_{\rm s}[1-X(z)]}}{\left(\frac{1}{\beta} + \lambda\tau_{\rm s}\right) \cdot [1 - X(z)]} \\ &\cdot \frac{(1-z) \cdot e^{-\frac{\lambda}{\mu}[1-X(z)]}}{e^{-\frac{\lambda}{\mu}[1-X(z)]} - z}. \end{aligned} \tag{12}$$

Remembering that the PGF is defined as

$$P_{\rm s}(z) = \sum_{k=0}^{\infty} P_k z^k \tag{13}$$

we can obtain the state probabilities P_k by calculating the Taylor series' coefficients of the function $P_{\rm s}(z)$

$$P_k = \frac{1}{k!}\frac{d^k}{dz^k} P_{\rm s}(z)\bigg|_{z=0}. \tag{14}$$

Notwithstanding these coefficients can be obtained in closed form through simple derivation operations, we preferred to evaluate such derivatives numerically since numerical evaluation has a lower computational complexity than calculating the closed-form expressions of the derivatives at $z = 0$.

E. Server Idle and Busy Times

Under server utilization factor $\rho < 1$, a $G/G/1$ queuing system will become empty infinitely often. This obviously remains true also for our $M^x/D/1/{\rm SET}$ model. Hence, using classical principles of renewal theory, we can identify independent and identically distributed (i.i.d.) "cycles" of the form

$$T_{\rm R}{}^{(n)} = T_{\rm I}{}^{(n)} + T_{\rm B}{}^{(n)} \tag{15}$$

where $T_{\rm B}{}^{(n)}$ is the nth busy period, (corresponding to the "delay busy period" in [27], which includes the setup time), and $T_{\rm I}{}^{(n)}$ is the nth idle period. In more detail, both sequences $T_{\rm B}{}^{(n)}$ and $T_{\rm I}{}^{(n)}$ can be demonstrated to be i.i.d. The average durations of idle and busy periods are given by

$$T_{\rm I} = E\left\{T_{\rm I}{}^{(n)}\right\} = \frac{1}{\lambda} \tag{16}$$

$$T_{\rm B} = \frac{1}{\lambda}\frac{\rho}{1-\rho} + \frac{\beta\tau_{\rm s}}{1-\rho}. \tag{17}$$

We can obtain $T_{\rm R}$ as follows:

$$T_{\rm R} = T_{\rm I} + T_{\rm B} = \frac{1 + \lambda\beta\tau_{\rm s}}{\lambda(1-\rho)}. \tag{18}$$

F. Network Performance Indexes

Starting from the stationary probabilities P_k obtained in Section III-D, as well as the idle and busy periods in Section III-E, we can easily derive a large set of network performance indexes.

1) Average Delay and Packet Loss Probability: The mean value $\bar{L}$ of packets in the queuing system can be obtained by specializing the general expressions in [27] to our case of deterministic service time and Zipf-distributed packet batches

$$\begin{aligned} \bar{L} &= \lim_{z\to 1} P_{\rm s}'(z) \\ &= \frac{2\lambda\beta\tau_{\rm s} + \lambda^2\beta^2{\tau_{\rm s}}^2 - \beta + \sum_{j=1}^{j_{\max}} \beta_j j^2}{2(1+\lambda\beta\tau_{\rm s})} \\ &\quad + \frac{\rho^2 - \beta + \sum_{j=1}^{j_{\max}} \beta_j j^2}{2(1-\rho)}. \end{aligned} \tag{19}$$

Using Little's law, the average waiting time $\bar{W}$ is

$$\begin{aligned} \bar{W} &= \frac{\bar{L}}{\lambda\beta} \\ &= \frac{2\tau_{\rm s} + \lambda\beta{\tau_{\rm s}}^2 - \frac{1}{\lambda} + \frac{1}{\lambda\beta}\sum_{j=1}^{j_{\max}} \beta_j j^2}{2(1+\lambda\beta\tau_{\rm s})} \\ &\quad + \frac{\rho^2 - \beta + \sum_{j=1}^{j_{\max}} \beta_j j^2}{2\lambda\beta(1-\rho)}. \end{aligned} \tag{20}$$

It is worth noting that both the function $P_{\rm s}(z)$ in (9) and the stationary probabilities P_k in (14) are referred to the $M^x/D/1/{\rm SET}$ queue with an infinite buffer. However, by assuming a low value of loss probability and similarly to [30], we can approximate the stationary probabilities of the finite buffer queuing system with the $\{P_0, P_1, \ldots, P_N\}$ probabilities of the $M^x/D/1/{\rm SET}$ queue. In more detail, the average value of packet loss probability can be expressed through the following approximation:

$$P_{\rm loss} = 1 - \sum_{k=0}^{N} P_k. \tag{21}$$

The approximation might be used also to recompute $\bar{L}$ and $\bar{W}$ for the finite buffer case. However, if $P_{\rm loss}$ is minute (as it actually turns out to be in most practical cases), (19) and (20) already provide a good approximation.

2) Energy Consumption: Recalling Fig. 3 and (1), we can express the average energy consumed in a renewal cycle as follows:

$$\tilde{\Phi} = \frac{(T_{\rm B} - \tau_{\rm on})}{T_{\rm R}} \cdot \Phi_{\rm a} + \frac{\tau_{\rm on}}{T_{\rm R}} \cdot \Phi_{\rm t} + \frac{T_{\rm I}}{T_{\rm R}} \cdot \Phi_{\rm idle} \tag{22}$$

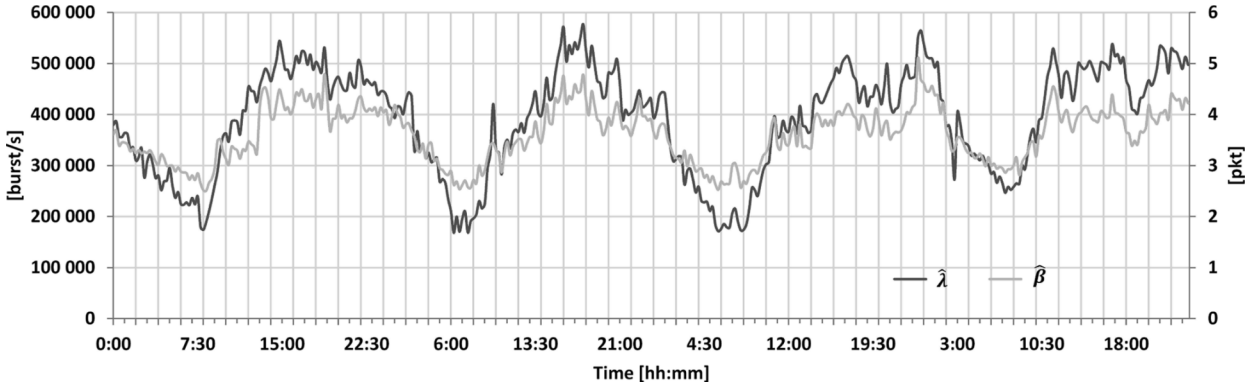

Fig. 4. Values of $\hat{\lambda}$ and $\hat{\beta}$ as extracted from the real traffic trace in [31].

TABLE II
POWER CONSUMPTIONS AND TRANSITION TIMES OF THE DEVICE'S C-STATES

C_x state	Φ_{idle}	τ_{on}
C_0	Active	Active
C_1	10 Watt	10 ns
C_2	8 Watt	100 ns

TABLE III
POWER CONSUMPTIONS AND FORWARDING CAPACITIES OF THE DEVICE'S P-STATES

P_y state	Φ_a	μ
P_3	50 Watt	650 kpkts/s
P_2	60 Watt	770 kpkts/s
P_1	70 Watt	890 kpkts/s
P_0	80 Watt	1010 kpkts/s

and by using (16)–(18) in (22)

$$\tilde{\Phi} = \left[1 + \frac{(\rho-1)(1+\lambda\tau_{\rm on})}{1+\beta\lambda\tau_{\rm s}}\right]\Phi_{\rm a} + \frac{\lambda(1-\rho)\tau_{\rm on}}{1+\beta\lambda\tau_{\rm s}}\Phi_{\rm t} + \frac{1-\rho}{1+\beta\lambda\tau_{\rm s}}\Phi_{\rm idle}. \quad (23)$$

IV. SINGLE PIPELINE MODEL VALIDATION

In order to validate the proposed model, we took the multicore Linux SW Router (SR) used in [21] as a term of comparison. This choice is mainly due to the fact that current commercial routers do not include AR and LPI capabilities, and only their nominal and/or maximum power consumptions are reported in the datasheets. In this section, we analyze the case with only one pipeline ($\Lambda = 1$). In Sections V and VI, we extend the single-pipeline case to that of multiple pipelines.

In more detail, the considered SW Router is equipped with several Gigabit Ethernet adapters with receive-side scaling (RSS) support [13]. Eight cores, placed in two Xeon 5550 processors, perform all packet forwarding operations in a fully parallel and independent way among themselves. As shown in Tables II and III, each processor core includes AR and LPI capabilities in terms of four available P-states and three C-states (including the C_0 one), respectively. Previous experimentations on SW router architectures [21] suggest to use the values indicated in Table II for the $\tau_{\rm on}$ parameter and to fix $\tau_{\rm r} = \frac{1}{\mu}$. In this scenario, we consider the behavior of each single core serving packets from reception interfaces. The parameters λ and β are the arrival rate and the average size of traffic batches processed by the considered core, respectively. For the sake of simplicity, we decided to show the validation results for a single processor core receiving traffic from a single Gigabit Ethernet interface with a reception buffer size equal to 512 packets and forwarding it toward another Gigabit Ethernet link. We performed the SW router experimentations and the proposed model estimation by using real-world traffic traces that are publicly available [31] and part of "A Day in the Life of the Internet" [32]. We used a 96-h-long traffic trace divided into sequential time windows of 15 min. Thus, for each time window, we obtained energy- and network-aware performance indexes both with the SW router and with the proposed model. The SW router measurements were performed by using the test bench composed by an Ixia N2X router tester [33] to reproduce traffic traces and to measure packet losses and latency times with high accuracy levels and an Agilent U2353A multifunction data acquisition (DAQ) device [34] to measure the processor power consumption. As far as the proposed model is concerned, for each time window, we used values calculated from the traffic trace for the parameters λ, β, ν, and $j_{\max}$.

As described in [1] and [35], typically the evolution of the incoming traffic load follows the classical night-and-day profile with high similarity between days. In [1], we noted that the minimum of the traffic typically appears during the first hours of the morning (from 03:00 to 06:00), while rush hours are during the day. Hence, in order to estimate the values of parameters for our model, we compute a moving average with the values of the previous days in the same time window. In more detail, for a generic day d and a window interval time $[t, t+\Delta t]$, we estimate the parameters λ and β with the moving average of the ones in the same interval of days $d-1, d-2, \ldots d-D$, where D is the number of samples in the moving average. In our evaluation, we set $D = 5$. Instead, the parameters ν and $j_{\max}$ were obtained, for each time window, by least-squares fitting of the Zipf distribution in (2) with the traffic trace; the final estimates are obtained by a moving average of the fitting results computed over the same window in the previous days.

In any case, in this paper we dot not focus on how the parameters are extracted from real traffic traces. At the state of the art, there are different methods and techniques that can be

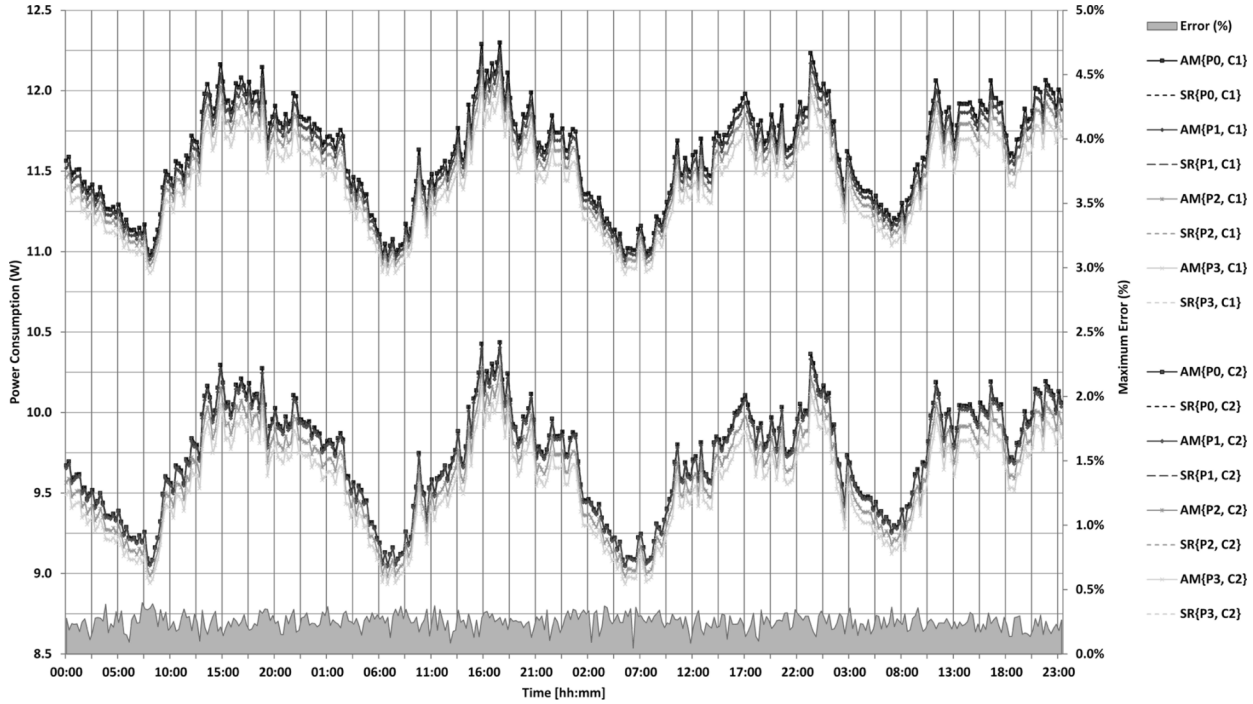

Fig. 5. Energy consumption estimated by the analytical model and measured on the SR, according to various configurations of P- and C-states.

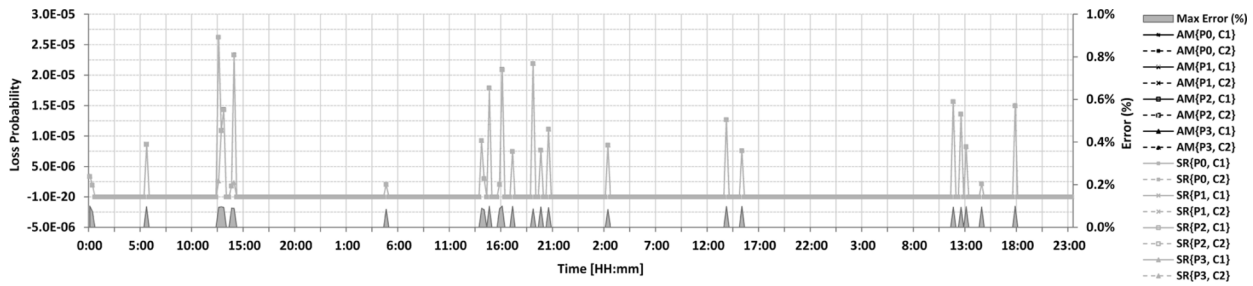

Fig. 6. Packet loss probability estimated by the AM and measured on the SR.

used to obtain these parameters with a given level of confidence. For example, a good methodology is based on Maximum likelihood estimators [36], [37]. Therefore, regardless of the specific parameter estimation method, our model evaluation shows an acceptable accuracy, which can be improved by using a more precise parameter estimator. The evolution of the traffic offered load over the time of the reference traffic trace is reported in Fig. 4 in terms of burst arrival rates and burst sizes. The minimum value of traffic loads is from 3:00 to 6:00, while rush hours occur at 11:00 and 14:00. It is interesting to underline how an increase in incoming traffic volume is due to the rise of both burst arrival rate and burst sizes. Fig. 5 reports the power consumption values estimated by the analytical model (AM), the values measured with the experimental test bench, and the maximum estimation error in each time window. The AM estimation was obtained with (23).

The results in Fig. 5 outline the good accuracy level provided by the model. Indeed, they suggest that selecting too deep standby states may cause a rise in power consumption. This is simply caused by the nonnegligible time $\tau_{\rm s}$ to wake up from the deepest C-state. When the probability of burst interarrival time being larger than $\tau_{\rm s}$ drops, the device enters low-power sleeping states more and more rarely and for shorter periods before waking up again. Figs. 6 and 7 show the average values of loss probability and packet latency times, respectively, for both the SR and the AM, as well as the relative estimation error. The AM estimates of latency times were obtained with (19) and (20), and loss probabilities were computed as in (21). Such results show that the proposed AM represents also network-aware performance indexes with a good accuracy level since the relative errors are lower than 0.1% for loss probabilities and lower than 2% for latency times. Regarding the AM complexity and execution times, the former depends linearly on the buffer size N, and the latter never exceeds 150 ms on an Intel Xeon 5560 running at 2.6 GHz.

V. Energy-Aware Load Balancing

Section IV described the model validation for a single pipeline case. In this section, we extend the model to a number of pipelines $\Lambda > 1$, and we insert it in an energy-aware load-balancing optimization problem. First, we introduce the definition of the optimization problem; then, we present numerical results referring to different tradeoff values and traffic volumes [38].

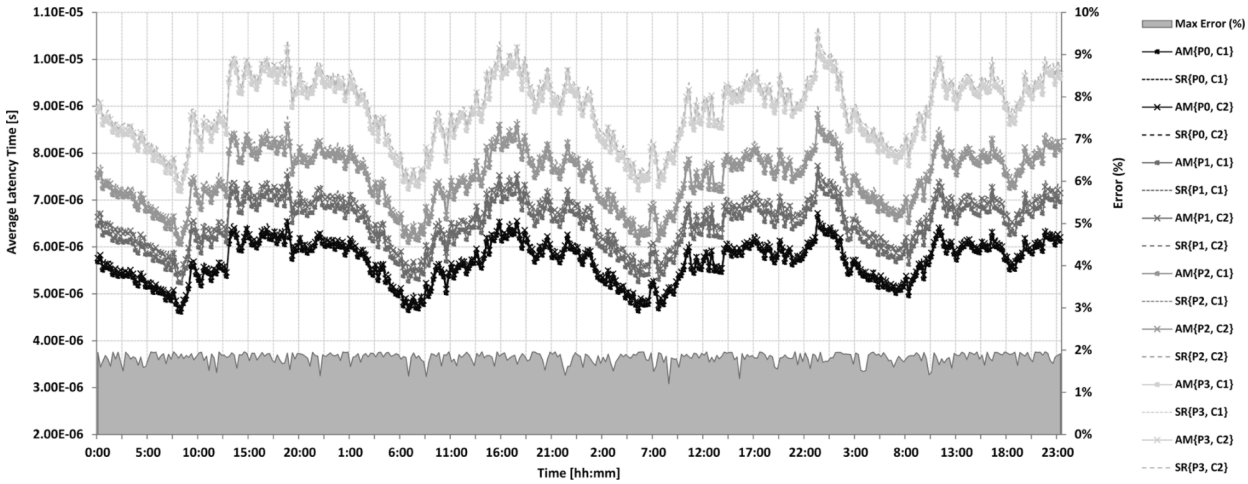

Fig. 7. Average packet latency estimated by the AM and measured on the SR.

We can define the average power consumption of our system as the sum of the contributions from the Λ parallel pipelines

$$\hat{\Phi} = \sum_{l=1}^{\Lambda} \tilde{\Phi}^{(l)}. \tag{24}$$

The average latency time experienced by a packet entering the system can be defined as

$$\hat{W} = \sum_{l=1}^{\Lambda} \frac{\lambda^{(l)}}{\hat{\lambda}} \bar{W}^{(l)}. \tag{25}$$

A. Optimization Problem Definition

Given the features of incoming traffic load (in terms of $\hat{\lambda}$, $\hat{\beta}$, and $\hat{\beta}_j$) and thresholds on the maximum values of both packet latency W^* and power consumption Φ^*, the objective of the load balancing criterion is to find the best values of $\lambda^{(i)}$, ${C_x}^{(i)}$, and ${P_y}^{(i)}$, $\forall\ i = 1, \ldots, \Lambda$, so that the system has the best tradeoff between network performance and energy consumption. Thus, we define our optimization problem as follows:

$$\begin{cases} \min\limits_{\substack{\lambda^{(i)} \\ {C_x}^{(i)} \in \{C_1, \ldots, C_X\} \\ {P_y}^{(i)} \in \{P_0, \ldots, P_Y\} \\ i=1 \ldots \Lambda}} & \gamma \frac{\hat{\Phi}}{\Phi^*} + (1-\gamma) \frac{\hat{W}}{W^*} \\ \hat{W} < W^*, \hat{\Phi} < \Phi^* \\ \sum_{l=1}^{\Lambda} \lambda^{(l)} = \hat{\lambda} \end{cases} \tag{26}$$

where the parameter γ ranges between 0 and 1 and represents the "tradeoff parameter," which modulates the minimization of power consumption with respect to the one of average packet latency. This optimization problem considers γ as a parameter and not as a variable. By varying the value of γ, the network operator would be able to adapt the problem to the desired targets. Indeed, for $\gamma = 0.00$, our optimization problem corresponds to the maximization of network performance for a given power consumption cap (as may be relevant, e.g., in the presence of networking equipment powered by renewable sources); on the other hand, setting $\gamma = 1.00$ to the minimization of the system power consumption constrained to a maximum value of average latency. Intermediate values can be used to reflect the relative importance attributed by the operator to the two goals.

Regarding the optimization problem, it is quite complex since we have a nonlinear objective function, which depends on both discrete (i.e., ${C_x}^{(i)}$, ${P_y}^{(i)}$, $\forall\ i = 1, \ldots, \Lambda$) and continuous (i.e., $\lambda^{(i)}$, $\forall\ i = 1, \ldots, \Lambda$) variables.

By taking into account that the number of pipelines Λ as well as that of available P- and C-states will typically be relatively low, our minimization strategy mainly consists of solving the problem for each available configuration of P- and C-states of the pipelines. In more detail, for each feasible combination of $\left\{\left({C_x}^{(1)}, {P_y}^{(1)}\right), \ldots, \left({C_x}^{(\Lambda)}, {P_y}^{(\Lambda)}\right)\right\}$, we find the best values of $\left\{\lambda^{(1)}, \ldots, \lambda^{(\Lambda)}\right\}$ minimizing the objective function and satisfying the constraints.

Moreover, by exploiting the last constraint in (26), we can express $\lambda^{(\Lambda)} = \hat{\lambda} - \sum_{l=1}^{\Lambda-1} \lambda^{(l)}$ and consequently reduce the number of variables.

Then, we simply try to find the minimum of the objective function by studying its partial derivatives in $\lambda^{(i)}$, $\forall\ i = 1, \ldots, \Lambda - 1$ inside the region satisfying the constraints, and on its frontier.

B. Analyzing the Tradeoff

In order to better understand and characterize the effects of the proposed optimization policy and the role of the tradeoff parameter γ, we performed some tests in the presence of variable incoming load.

In more detail, we consider a packet processing engine with $\Lambda = 4$ pipelines with the same configuration for the single-pipeline case described in Section IV. Each pipeline corresponds to a processor core and includes the same AR and LPI capabilities as used in the case of a single pipeline.

Considering the real traffic traces in Fig. 4, we decided to fix $\hat{\beta} = 4$, while we increased the value of $\hat{\lambda}$ from 1 kpkt/s to 2.5 Mpkt/s (which, in our case, roughly corresponds to the threshold beyond which the optimization constraints cannot be satisfied).

The optimization problem has been solved for various values of the tradeoff parameter corresponding to $\gamma = 0.00, 0.25, 0.50,$

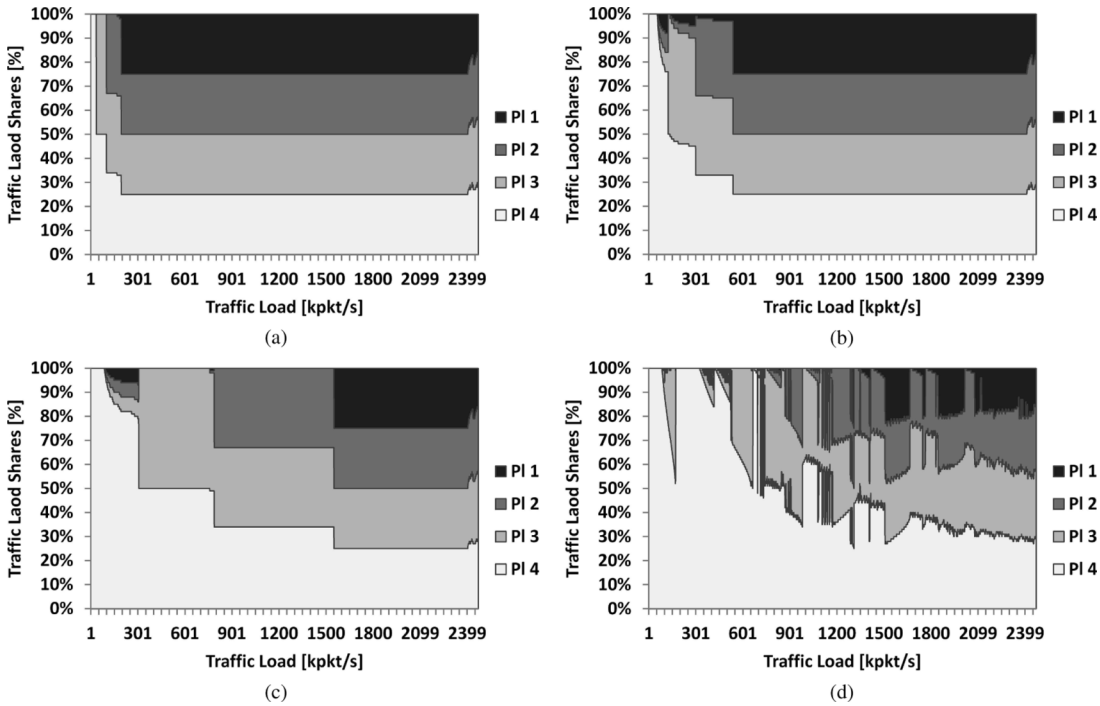

Fig. 9. Optimal load shares for each Pl and for $\gamma = \{0.25, 0.50, 0.75, 1.00\}$ according to increasing traffic volumes. The type of graph used is the stacked area chart, where each area refers (in %) to the optimal load shares among the pipelines. (a) $\gamma = 0.25$. (b) $\gamma = 0.50$. (c) $\gamma = 0.75$. (d) $\gamma = 1.00$.

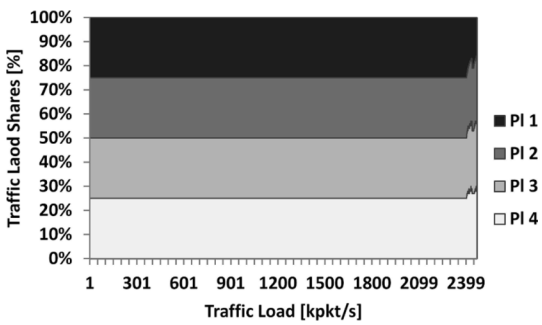

Fig. 8. Optimal load shares for each pipeline (Pl) and for $\gamma = 0.00$ according to increasing traffic volumes. The type of graph used is the stacked area chart, where each area refers (in %) to the optimal load shares among the pipelines.

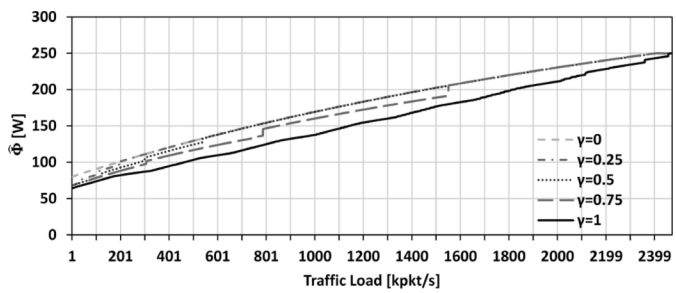

Fig. 10. Average power consumption of the packet processing engine with respect to γ and increasing values of $\hat{\lambda}$.

0.75, and 1.00. The maximum latency W^* has been fixed to 20 μs, and the constraint on power consumption Φ^* to 250 W. Figs. 8 and 9 show the optimal shares $\left\{\lambda^{(1)}, \ldots, \lambda^{(4)}\right\}$ of incoming traffic load for each pipeline with respect to different values of γ. Figs. 10 and 11 report the estimated power consumptions and the packet latency times, respectively, in the optimal configurations. Figs. 12 and 13 show how many pipelines are working in the available P- and C-states in the cases $\gamma = 0.00$ and $\gamma = 0.75$, respectively.

By observing Figs. 8 and 9, we can outline how, in the case of minimization of the latency times constrained to the energy consumption (i.e., $\gamma = 0.00$), the optimal policy suggests to uniformly divide the incoming load among the pipelines. Only for the highest load volumes ($\hat{\lambda} > 2.4$ Mpkt/s), this fairness is not maintained. In fact, in order to satisfy the power consumption constraint, the optimization policy maintains three pipelines with P_0 and C_1 and reduces the energy consumption of the whole engine by decreasing the performance of pipeline 1. Accordingly, the load balancer reduces the load share incoming to this pipeline.

On the contrary, when we minimize the power consumption for a given threshold on maximum latency times (i.e., $\gamma = 1.00$), the load balancer tries to concentrate as much traffic volume as possible into few pipelines. For instance, and with reference to Fig. 9(d), the load balancer redirects traffic only to pipeline 4 at very low incoming volumes. When a change is needed in the P- and C-state configuration of pipeline 4 to satisfy the network performance constraints, the optimization policy decides to delay this configuration change and to use also other (few) pipelines. However, by further increasing the incoming traffic load, the configuration change on pipeline 4 becomes soon more energy-efficient, and the largest part of the load returns to this pipeline. When $\hat{\lambda} > 1.5$ Mpkt/s, the optimization policy starts to distribute traffic among pipelines in a fairer way in order to satisfy the W^* constraint.

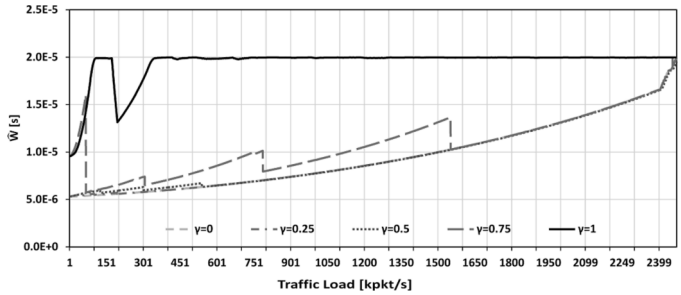

Fig. 11. Average packet latency times of the processing engine with respect to γ and increasing values of $\hat{\lambda}$.

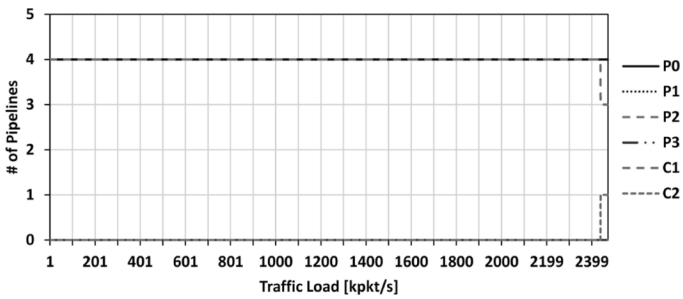

Fig. 12. Number of pipelines working in P_y and in C_x state for $\gamma = 0.00$.

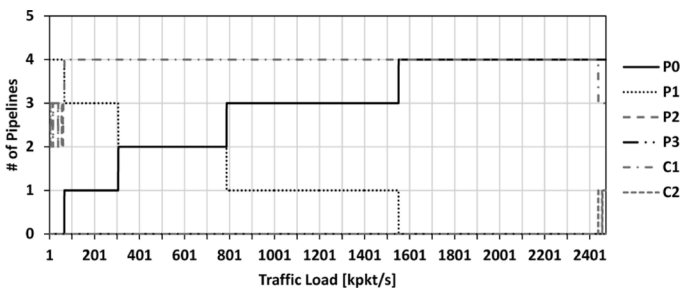

Fig. 13. Number of pipelines working in P_y and in C_x state for $\gamma = 0.75$.

Regarding energy consumption and average latency times, the case $\gamma = 1.00$ exhibits a nearly linear behavior in $\hat{\Phi}$ with respect to $\hat{\lambda}$, while $\hat{W}$ is almost equal to W^* for the largest part of $\hat{\lambda}$ values. This behavior is sensibly different with respect to the case $\gamma = 0.00$, where $\hat{\Phi}$ increases with a concave trend according to $\hat{\lambda}$, and $\hat{W}$ values remain much lower than W^*. As far as the other values of γ are concerned [see Fig. 9(a)–(c)], the optimization policy roughly behaves as the minimization of power consumption ($\gamma = 1.00$) at low traffic volumes, and as the minimization of packet latency ($\gamma = 0.00$) at higher loads. The macroscopic role of the tradeoff parameter γ appears to be moving the point where the optimization policy switches between the minimization of power consumption and the maximization of network performance: As γ increases, the region with unbalanced traffic share enlarges. This role is also evident in Fig. 10, where the power consumptions of the cases $\gamma = 0.25$, 0.50, and 0.75 start by agreeing with the $\gamma = 1.00$ curve, and by increasing $\hat{\lambda}$, they end up, one by one, by meeting the $\gamma = 0.00$ values. As γ rises, such meeting point happens at higher traffic volumes. In addition, by observing Figs. 12 and 13, we can outline also that the P- and C-states transitions become more

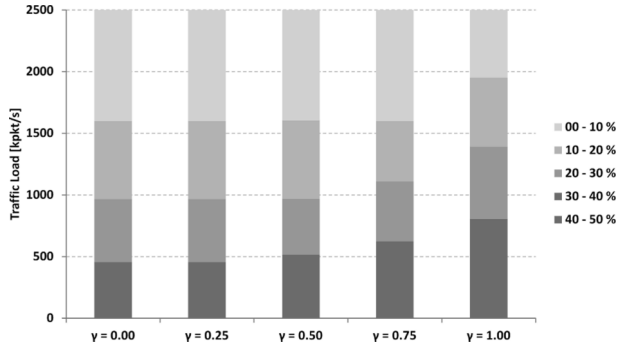

Fig. 14. Average energy saving (in %) for a high-end IP router with respect to γ and increasing values of $\hat{\lambda}$. Router consumption data from Tucker *et al.* [8] and Neilson [9].

frequent according to the increase of γ. To highlight the energy-efficiency impact of the optimization performed on the packet processing engine with respect to the overall router consumption, Fig. 14 shows the energy saving (in %) for a high-end IP router with respect to γ and increasing values of $\hat{\lambda}$. These values of energy saving are computed based on the consumption data breakdown of Tucker *et al.* [8] and Neilson [9], already mentioned in Section I. The computed values consider only the energy saving deriving from packet processing engines with the energy-aware analytical model enabled. By taking into account a more efficient use of other components (e.g., fans and power supplies), the overall saved energy would be even greater.

VI. Energy-Aware Load Balancing Evaluation

In this section, we examine further results of some tests in order to understand the effects of the proposed optimization policy and the role of the tradeoff parameter γ in the multiple-pipeline case. Hence, we provide a performance evaluation of the optimization procedure described in Section V-A using the real traffic traces introduced in Section IV.

Figs. 15 and 16 show the estimated values for $\hat{\Phi}$ and $\hat{W}$, respectively, in the optimal configuration with respect to the traffic trace time windows and different values of the tradeoff parameter γ. In the same scenario, Figs. 17 and 18 show how many pipelines are using a certain P- and C-states pair, respectively.

These figures clearly outline how the optimization policies for $\gamma = 0.00$, 0.25, and 0.50 provide almost the same results. This behavior occurs because the offered load in the traffic trace we considered is always relatively high (greater than about 600 kpkt/s almost everywhere in the trace); therefore, as discussed in Section V-B, the optimization policy behaves like the pure minimization of packet latency for small values of γ (up to 0.50), and it provides the same configuration of P- and C-states and the same traffic shares among the pipelines in all cases. However, in the case of $\gamma = 1.00$, the optimization policy allows saving about 12% of energy with respect to $\gamma = 0.00$. On the other hand, with $\gamma = 1.00$, the average packet latency time is always close to the W^* value. Finally, for $\gamma = 0.75$, we have an increase in energy saving of 2.5% with respect to $\gamma = 0.00$, and the $\hat{W}$ values appear to be a bit higher (max 5 μs) than at $\gamma = 0.00$, especially during low-load periods (from 00:00 to 09:00). For each case evaluated, the average packet latency was

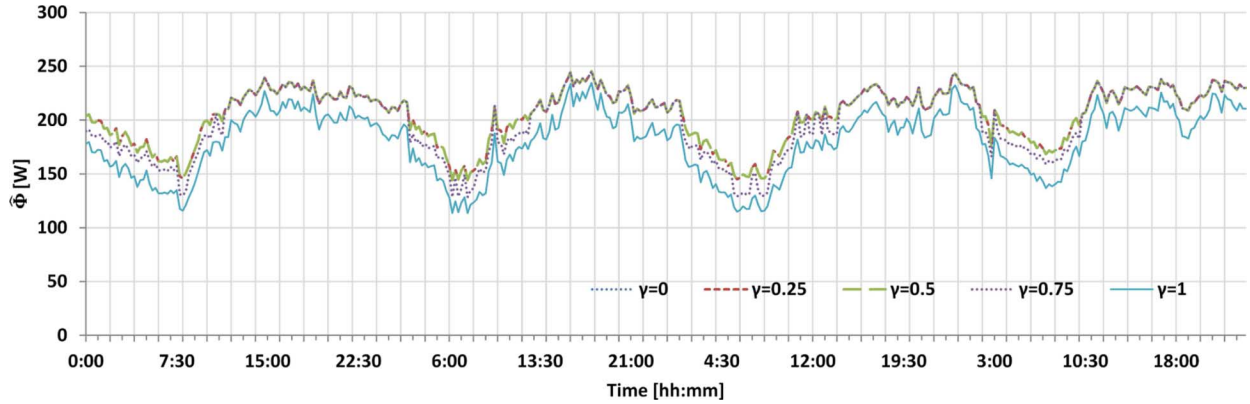

Fig. 15. Power consumption $\hat{\Phi}$ for various values of γ with respect to the real traffic trace in [31].

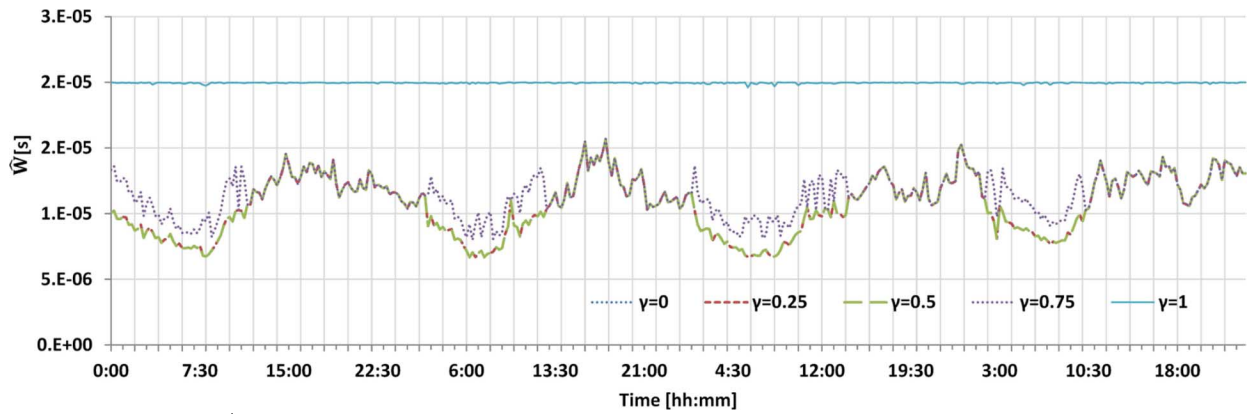

Fig. 16. Average latency times $\hat{W}$ for various values of γ with respect to the traffic trace in [31].

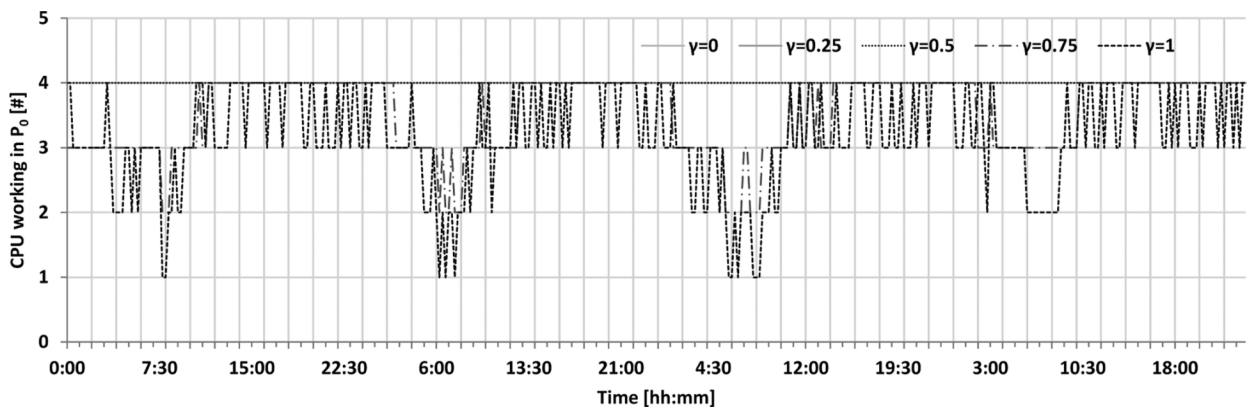

Fig. 17. Number of pipelines working in P_0 for various values of γ with respect to the traffic trace in [31].

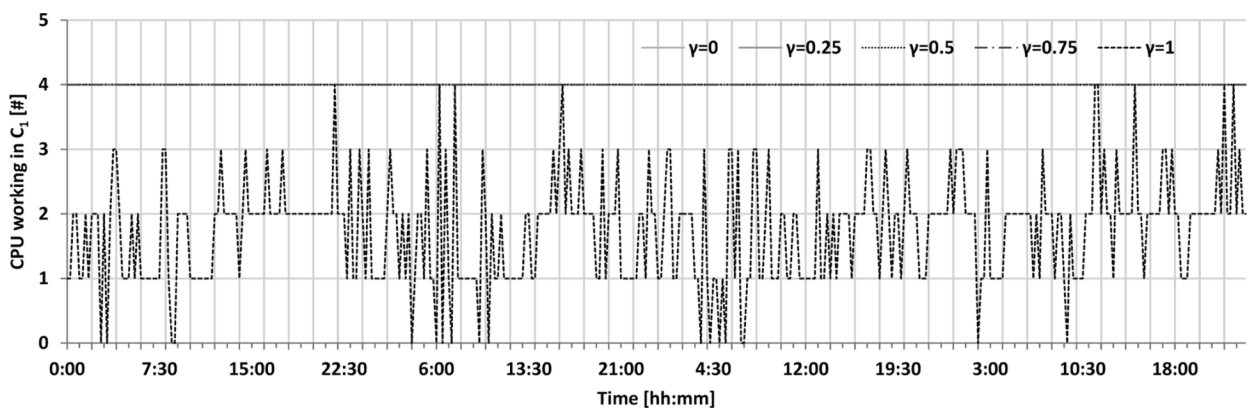

Fig. 18. Number of pipelines working in C_1 for various values of γ with respect to the traffic trace in [31].

always less than 20 μs. It is worth noting that if similar values of packet latency are maintained as a constraint, no timeout would be triggered in TCP connections. On the other hand, we have seen that packet loss probabilities are very low with

well-tuned control system parameters.[4] For these reasons, we have neglected the overhead caused by packets being resent.

VII. Conclusion

In this paper, we considered energy-aware network devices (e.g., routers, switches, etc.) able to trade their energy consumption for packet forwarding performance by means of both low power idle and adaptive rate schemes.

We proposed a novel analytical model able to capture the impact of power management capabilities on network performance metrics. The analytical framework considers stochastic incoming traffic at the packet level with LRD properties. On the basis of the analytical model, we choose the parameters characterizing the joint usage of AR and LPI energy-aware capabilities by optimizing the desired tradeoff between energy consumption and QoS while at the same time enforcing the satisfaction of given upper bounds on both. Since the performance and cost indicators used in the optimization depend on incoming traffic volumes and statistical features (notably, burst interarrival time and average burst length), we repeat the optimization periodically under updated estimations of these quantities. The modeling and control framework has been validated experimentally by using a Linux-based open software router with AR and LPI primitives under traffic generated by real-world traces; the results demonstrate how the proposed model can effectively represent energy- and network-aware performance indexes.

We focused on state-of-the-art packet processing engines, which generally represent the most energy-consuming components of network devices, and which are often composed of a number of parallel pipelines to "divide and conquer" the incoming traffic load. Our goal was to control both the power configuration of pipelines, and the best way to distribute traffic flows among them, in order to optimize the tradeoff between energy consumption and network performance indexes.

The obtained results show that the proposed optimization policy for low traffic volumes roughly corresponds to the minimization of energy consumption constrained to a maximum packet latency. For higher values, the same policy starts to maximize network performance for a given energy-cap. By tuning the tradeoff parameter in the proposed objective function, we can control at which incoming load the policy switches between the two behaviors. Further work along similar lines to investigate other models and control strategies is still ongoing within the framework of some of the research projects that contributed to the results in this paper [39]–[41].

References

[1] R. Bolla, R. Bruschi, A. Carrega, F. Davoli, D. Suino, C. Vassilakis, and A. Zafeiropoulos, "Cutting the energy bills of Internet service providers and telecoms through power management: an impact analysis," *Comput. Netw.*, vol. 56, no. 10, pp. 2320–2342, 2012.

[2] C. Bianco, F. Cucchietti, and G. Griffa, "Energy consumption trends in the next generation access network—a telco perspective," in *Proc. 29th INTELEC*, Rome, Italy, Sep.–Oct. 2007, pp. 737–742.

[4]To be even more conservative against the possibility of temporary surges in peak traffic load, an explicit constraint on packet loss probability can always be introduced in the optimization problem (as was done in the single-pipeline case).

[3] The Climate Group—GeSI, "SMART 2020: Enabling the low carbon economy in the information age," 2008 [Online]. Available: http://www.smart2020.org/_assets/files/02_Smart2020Report.pdf

[4] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing network energy consumption via sleeping and rate-adaptation," in *Proc. 5th USENIX NSDI*, San Francisco, CA, USA, 2008, pp. 323–336.

[5] R. Bolla, F. Davoli, R. Bruschi, K. Christensen, F. Cucchietti, and S. Singh, "The potential impact of green technologies in next-generation wireline networks: Is there room for energy saving optimization?," *IEEE Commun. Mag.*, vol. 49, no. 8, pp. 80–86, Aug. 2011.

[6] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy efficiency in the future Internet: A survey of existing approaches and trends in energy-aware fixed network infrastructures," *IEEE Commun. Surveys Tut.*, vol. 13, no. 2, pp. 223–244, 2nd Quart., 2011.

[7] L. Chiaraviglio, M. Mellia, and F. Neri, "Minimizing ISP network energy cost: Formulation and solutions," *IEEE/ACM Trans. Netw.*, vol. 20, no. 2, pp. 463–476, Apr. 2012.

[8] R. Tucker, R. Parthiban, J. Baliga, K. Hinton, R. Ayre, and W. Sorin, "Evolution of WDM optical IP networks: A cost and energy perspective," *J. Lightw. Technol.*, vol. 27, no. 3, pp. 243–252, Feb. 2009.

[9] D. Neilson, "Photonics for switching and routing," *IEEE J. Sel. Topics Quantum Electron.*, vol. 12, no. 4, pp. 669–678, Jul.–Aug. 2006.

[10] Netlogic, Irvine, CA, USA, "The Netlogic XLP processor family," [Online]. Available: http://www.netlogicmicro.com/Products/MultiCore/XLP.asp

[11] "The NetFPGA Project," [Online]. Available: http://www.netfpga.org

[12] S. Han, K. Jang, K. Park, and S. Moon, "PacketShader: A GPU-accelerated software router," in *Proc. ACM SIGCOMM*, New Delhi, India, Aug.–Sep. 2010, vol. 40, pp. 195–206.

[13] Z. Yi and P. Waskiewicz, "Enabling Linux network support of hardware multiqueue devices," in *Proc. Linux Symp.*, Ottawa, ON, Canada, Jun. 2007, vol. 2, pp. 305–310.

[14] W. Shi, M. MacGregor, and P. Gburzynski, "Load balancing for parallel forwarding," *IEEE/ACM Trans. Netw.*, vol. 13, no. 4, pp. 790–801, Aug. 2005.

[15] A. Smiljanic, "Rate and delay guarantees provided by CLOS packet switches with load balancing," *IEEE/ACM Trans. Netw.*, vol. 16, no. 1, pp. 170–181, Feb. 2008.

[16] Y. Bejerano, S.-J. Han, and L. Li, "Fairness and load balancing in wireless LANs using association control," *IEEE/ACM Trans. Netw.*, vol. 15, no. 3, pp. 560–573, Jun. 2007.

[17] S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic load balancing without packet reordering," *Comput. Commun. Rev.*, vol. 37, pp. 51–62, Mar. 2007.

[18] R. Bolla, R. Bruschi, A. Carrega, and F. Davoli, "Green network technologies and the art of trading-off," in *Proc. 30th IEEE INFOCOM Workshops*, Shanghai, China, Apr. 2011, pp. 301–306.

[19] R. S. Martin and J. Knight, "Power-profiler: Optimizing ASIC's power consumption at the behavioral level," in *Proc. 32nd ACM/IEEE DAC*, San Francisco, CA, USA, Jun. 1995, pp. 42–47.

[20] Hewlett-Packard Corp., Intel Corp., Microsoft Corp., Phoenix Technologies Ltd., and Toshiba Corp, "Advanced configuration and power interface specification," Apr. 2010 [Online]. Available: http://www.acpi.info/DOWNLOADS/ACPIspec40a.pdf,

[21] R. Bolla, R. Bruschi, and A. Ranieri, "Green support for PC-based software router: Performance evaluation and modeling," in *Proc. 45th IEEE ICC*, Dresden, Germany, 2009, pp. 2200–2205.

[22] R. Bolla, R. Bruschi, F. Davoli, and A. Ranieri, "Energy-aware performance optimization for next-generation green network equipment," in *Proc. 2nd ACM SIGCOMM PRESTO*, Barcelona, Spain, Aug. 2009, pp. 49–54.

[23] V. Paxton and S. Floyd, "Wide-area traffic: The failure of poisson modeling," *IEEE/ACM Trans. Netw.*, vol. 3, no. 3, pp. 226–244, Jun. 1995.

[24] W. Willinger, V. Paxson, and M. Taqqu, *Self-Similarity and Heavy Tails: Structural Modeling of Network Traffic*. Cambridge, MA, USA: Birkhauser, 1998, pp. 27–53.

[25] P. Salvador, A. Pacheco, and R. Valadas, "Modeling IP traffic: Joint characterization of packet arrivals and packet sizes using BMAPs," *Comput. Netw.*, vol. 44, pp. 335–352, Feb. 2004.

[26] A. Klemm, C. Lindemann, and M. Lohmann, "Modeling IP traffic using the batch markovian arrival process," *Perform. Eval.*, vol. 54, pp. 149–173, Oct. 2003.

[27] G. Choudhury, "An $M^x/G/1$ queueing system with a setup period and a vacation period," *Queueing Syst., Theory Appl.*, vol. 36, pp. 23–38, Nov. 2000.

[28] H. Tijms, *A First Course in Stochastic Models*. Chichester, U.K.: Wiley, 2003.
[29] B. Doshi, "A note on stochastic decomposition in a $GI/G/1$ queue with vacations or set-up times," *J. Appl. Probab.*, vol. 22, no. 2, pp. 419–428, Jun. 1985.
[30] H. Kim and N. Shroff, "Loss probability calculations and asymptotic analysis for finite buffer multiplexers," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 755–768, Dec. 2001.
[31] Working Group Traffic Archive, "Sample Point F," 2008 [Online]. Available: http://mawi.nezu.wide.ad.jp/mawi/samplepoint-F/20080318
[32] "A day in the life of the Internet project," [Online]. Available: http://www.caida.org/projects/ditl
[33] CAIDA, La Jolla, CA, USA, "The Ixia IxN2X router tester," Ixia [Online]. Available: http://www.ixiacom.com/products/ixn2x/index.php
[34] Agilent, Santa Clara, CA, USA, "Agilent U2353A Multifunction DAQ," [Online]. Available: http://cp.literature.agilent.com/litweb/pdf/5989-9923EN.pdf
[35] ECONET Project, "Deliverable 2.1," 2011.
[36] H. Bauke, "Parameter estimation for power-law distributions by maximum likelihood methods," *Eur. Phys. J. B, Condensed Matter Complex Syst.*, vol. 58, no. 2, pp. 167–173, 2007.
[37] D. Felletti, "An algorithm for maximum likelihood estimation of Zipf's law," University of Milano-Bicocca, Milan, Italy, Nov. 2010 [Online]. Available: http://hdl.handle.net/10281/17659
[38] R. Bolla and R. Bruschi, "Energy-aware load balancing for parallel packet processing engines," in *Proc. 1st IEEE GREENCOM*, Sep. 2011, pp. 105–112.
[39] "Low Energy Consumption NETworks (ECONET) project," 2010 [Online]. Available: http://www.econet-project.eu
[40] "Energy eFFIcient teChnologIEs for the Networks of Tomorrow (EFFICIENT) project," 2010 [Online]. Available: http://www.tnt.dist.unige.it/efficient
[41] "Greening the Network (GreenNet) project," 2012 [Online]. Available: http://www.tnt.dist.unige.it/greennet

Raffaele Bolla (M'91) was born in Savona, Italy, in 1963. He received the "Laurea" degree in electronic engineering and Ph.D. degree in telecommunication engineering from the University of Genoa, Genova, Italy, in 1989 and 1994, respectively.

From 1996 to 2004, he was a Researcher with the Department of Communications, Computer and Systems Science (DIST), University of Genoa. Since 2004, he has been an Associate Professor with the University of Genoa, where he teaches a course in telecommunication networks and telematics. He has authored or coauthored over 150 scientific publications in international journals and conference proceedings. He has been the Principal Investigator in many projects in the field of telecommunication networks. His current research interests are in resource allocation, call admission control and routing in multiservice IP networks, multiple access control, resource allocation and routing in both cellular and ad hoc wireless networks, and energy-efficient networking.

Roberto Bruschi (M'09) received the M.Sc. degree in telecommunication engineering and Ph.D. degree in electronic engineering from the University of Genoa, Genova, Italy, in 2002 and 2006, respectively.

Since 2009, he has been a Researcher with the National Inter-University Consortium for Telecommunications, University of Genoa Research Unit. He is the Technical Coordinator Assistant in the ECONET project, and the Principal Investigator in the GreenNet project. He has coauthored about 50 scientific papers in international journals, book chapters, and international conference proceedings. His main research interests include green networking, software routers, and multiple-play networks.

Dr. Bruschi has been a technical committee member of many international conferences. In 2009, he won the Best Paper Award at the Next-Generation Networking Symposium of the IEEE ICC, and in 2010 at the IEEE Green Communications Workshop colocated with GLOBECOM 2010.

Alessandro Carrega (S'11) was born in Novi Ligure, Italy. He received the "Laurea" degree (equivalent to M.Sc.) in computer engineering at the University of Genoa, Genova, Italy, in 2008, and is currently pursuing the Ph.D. degree within the Telecommunication Networks and Telematics Lab. (TNT), Department of Electrical, Electronic and Telecommunication Engineering, and Naval Architecture (DITEN), University of Genoa.

In 2011, he was a Visiting Ph.D. Scholar with Portland State University (PSU), Portland, OR, USA, under the supervision of Prof. Suresh Singh. He has coauthored several papers in international conference proceedings. His main research interests include green networking, software routers, network virtualization, and OpenFlow.

Mr. Carrega is a member of CNIT, the Italian Inter-University Consortium for Telecommunications. In 2010, he won the Best Paper Award at the 3rd International Workshop on Green Communications (GreenCom10) colocated with the IEEE GLOBECOM Conference.

Franco Davoli (M'90–SM'99) received the "Laurea" degree in electronic engineering from the University of Genoa, Genova, Italy, in 1975.

Since 1990, he has been a Full Professor of telecommunication networks with the University of Genoa, where he is with the Department of Electrical, Electronic and Telecommunication Engineering, and Naval Architecture (DITEN). In 2004 and 2011, he was a Visiting Erskine Fellow with the University of Canterbury, Christchurch, New Zealand. He has coauthored over 300 scientific publications in international journals, book chapters and conference proceedings. His current research interests are in bandwidth allocation, admission control and routing in multiservice networks, wireless mobile and satellite networks, multimedia communications and services in distributed computing environments, and energy-efficient networking.

Prof. Davoli has been Investigator in a large number of projects and has served in several positions in the Italian National Consortium for Telecommunications (CNIT). He was a co-founder and the Head, for the term 2003–2004, of the CNIT National Laboratory for Multimedia Communications, Naples, Italy, and Vice-President of the CNIT Management Board for the term 2005–2007.