Modelling and Real-Trace-Based Evaluation of Static and Dynamic Coalescing for Energy Efficient Ethernet

Angelos Chatzipapas
angelos.chatzipapas@imdea.org

Vincenzo Mancuso
vincenzo.mancuso@imdea.org

Institute IMDEA Networks, and University Carlos III of Madrid
Madrid, Spain

ABSTRACT

The IEEE Standard 802.3az, namely Energy Efficient Ethernet (EEE), has been recently introduced to reduce the power consumed in LANs. Since then, researchers have proposed various traffic shaping techniques to leverage EEE in order to boost power saving. In particular, packet coalescing is a promising mechanism which can be used on top of EEE to tradeoff power saving and packet delay. In this paper, we analyze the interesting and special case of 1000Base-T EEE links, in which power saving operations are triggered only when links are inactive in both transmission directions. We are the first to provide an analytical model for EEE 1000Base-T which accounts for the bidirectional nature of LAN traffic. Our model allows to compute the power saving achieved by EEE, with and without packet coalescing, by using a few significant traffic descriptors. Furthermore, we use real traffic traces to investigate on the performance of static as well as dynamic coalescing schemes. Our results show that dynamic coalescing does not significantly outperform static coalescing in terms of power save and delay.

Categories and Subject Descriptors

C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks—*Ethernet*; G.3 [**Probability and Statistics**]: Queueing Theory

Keywords

EEE; Bidirectional Gigabit Links; Dynamic Coalescing

1. INTRODUCTION

During the past years a lot of effort has been invested to increase processing, communication, switching speed and data storage with little effort to optimize the power consumption. According to [1], about 14 TWh were consumed in 2005 by the telecom core network in EU-25[1] and the yearly consumption is expected to increase to about 30 TWh by 2020. Although this power consumption is useful for the human beings, it is also potentially harmful for our environment since it produces an augmented amount of CO_2 emissions and highly contributes to the greenhouse effect. The current threat to the environment could turn into a much more serious threat in the near future, since there is a growing demand of new generation devices that require connection to the Internet (such as televisions, white goods, etc.). In addition, existing network connected devices are now increasing their bandwidth demands (e.g., Web servers, databases, etc.). Indeed, the Internet traffic might grow with the number of data centers in the network and the number of users that demand higher amounts of traffic such as bigger files, videos, TV over IP etc. Hence, as the data traffic demand rises, especially in developing countries, more and more energy consumption is expected for networking.

In order to protect the environment and obtain lower service cost, Internet Service Providers and Network Operators are currently deploying new strategies to reduce energy consumptions. In this context, our work investigates on the recently approved Energy Efficient Ethernet (EEE) standard for power saving in Local Area Networks. Indeed, according to [2], the authors estimate significant reductions of about 4 TWh per year over one billion devices.

Legacy Ethernet is a power-unaware standard which consumes a constant amount of power independently from the actual traffic flowing through the wires. However, low speed Ethernet cards consume about 200 mW, which is not a significant consumption considering that a server or a home PC consumes tens to thousands of Watts. Therefore, so far Ethernet power saving strategies did not rise the interest of researchers and developers, due to the irrelevance of potential savings for low speed connections. In contrast, new high speed Gigabit interface cards may consume up to 20 W [3] which makes reasonable the introduction of a power saving mechanism. In fact, taking into account that the amount of Web Hosting Centers and server farms has been extremely increased due to the new trends and services (YouTube, Facebook, Twitter etc.), there are now billions of running interfaces that consume a constant amount of power. In addition, Ethernet links are basically inactive most of the time. Therefore, a new power aware Ethernet standard (standardized late 2010) was introduced to minimize the power consumption of the links when low traffic is present, namely IEEE 802.3az, or EEE [4].

While some effort has been put in understanding the behavior of EEE links where power saving can be activated independently in each traffic direction, e.g., [5, 6, 7], in this paper, we are the first to present an analytical model for bidirectional EEE links, e.g., EEE links in which power saving operations can only be activated when there is no traffic in both link directions. The latter (namely, the *bidirectional EEE case*) is a very relevant case, since the EEE standard adopts this bidirectional behavior for 1000Base-T

[1]The first 25 countries that joined the European Union.

e-Energy'13, May 21–24, 2013, Berkeley, California, USA.

cards, which are the most commonly adopted and diffused gigabit network cards, as of today. Notably, packet coalescing techniques have been proposed to boost EEE performance when traffic is not scarce and packet arrivals have short spacing. The basic idea behind coalescing is to aggregate packets in a buffer of limited size until either the buffer is full or a timeout expires.

We propose a model that uses simple statistical parameters (such as mean interarrival time and its variance) to estimate the power consumption of a bidirectional EEE link over time, and the packet delay incurred in crossing the link when coalescing techniques are adopted.

We collected real traces from a large web hosting center and extracted from them the traffic parameters needed by our model. We also used real traces to validate our model in terms of EEE power saving and packet delay by simulating the EEE and packet coalescing behaviors with real input traffic, using a modified ns-3 simulator [8]. In line with other studies focusing on EEE, e.g., [5, 9], we show that, without coalescing, EEE enables non-negligible power saving only when the offered traffic is rather low (few percents of the link capacity) and packet arrivals are bursty.

Another fundamental contribution of our work consists in the performance evaluation of packet coalescing strategies for EEE. Not only we evaluate the power saving enhancements achievable by means of static coalescing approaches, but we also propose dynamic strategies to adapt the coalescing parameters to the traffic characteristics. Our performance analysis shows that both the size of the coalescing queue and the duration of the coalescing timeout should adapted to the offered traffic. However, we show that a dynamic coalescing approach can be used, which seamlessly adapts to any traffic conditions, and achieves nearly optimal results in terms of power saving and packet delay. Nonetheless, our thorough investigation shows that also static coalescing can achieve nearly optimal results, thus questioning the importance of exploring more complex approaches based on run time adaptation of the coalescing parameters.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 presents the EEE standard and describes the behavior of 1 *Gbps* EEE links. Section 4 describes an analytical model for the estimation of power consumption and packet delay in bidirectional EEE links. Section 5 introduces dynamic coalescing algorithms to adapt the coalescing parameters to the time-varying traffic conditions. In Section 6 we use real-trace-based simulations to validate our model and present an extensive performance evaluation of EEE with and without coalescing schemes. Section 7 summarizes and concludes the paper.

2. RELATED WORK

Modeling of EEE. Various analytical models exist in the literature for *unidirectional* links. In [5] the authors propose an analytical model that allows to compute fast the potential EEE power saving and it performs very well for EEE links with no coalescing, using simple statistical parameters for unidirectional traffic. In [10], using parameters such as the packet arrival time and the service rate of the *coalescer*, the model is able to compute the mean queue length, the mean packet delay and the delay for the downstream queue for 10 *Gbps* links. A two state analytical model for 10GBase-T links is presented in [6] that estimates the power consumption of EEE links. This is a quick modeling tool, but it is not very accurate in case of small transition intervals, since it divides the time into discrete intervals equal to multiples of frame transmission time. Herrería-Alonso *et al.* [7] propose and analyze a model for both legacy EEE and burst transmission with 10 *Gbps* cards. Their model estimates the power saving using the arrival rate for Poisson traffic and the average service rate. They also propose a model with GI/G/1 queues for both frame and burst transmissions. The model allows to compute the average delay of packets and the power saving of the link using Poisson and deterministic traffic, but it is specifically designed for the case of 10 *Gbps* links and thus it cannot be used to estimate the power saving of the widely used 1 *Gbps* links [9].

EEE performance evaluation. A few number of EEE evaluations and extension proposals have appeared during the last few years to study and improve EEE's performance. Reviriego *et al.* proved initially the inefficiency of EEE by simulating the standard on ns-2 for 100Base-T, 1000Base-T and 10GBase-T links [11]. In [12] the authors provide a first evaluation on newly released EEE NIC cards for 1 *Gbps* links. They measure the power consumption of the cards with real traffic and prove that: (i) the power consumption during transitions is similar to the power consumption in "Active" state and, (ii) great power saving can be achieved but for very low loads, reporting saving up to 30% for 100 $Mbps$ links and up to 70% for 1 *Gbps* links. We have previously shortly reported on traffic measurements and potential power saving under EEE gigabit links in [13].

Coalescing in EEE links. One of the first evaluations of packet coalescing for EEE is presented in [2] for 10 *Gbps* Ethernet links. The results show that packet coalescing outperforms legacy EEE in terms of power consumption and it overcomes the major problem of EEE, namely the overhead due to protocol state transitions (which correspond to hardware operational states). However, EEE introduces additional delay for the packets to cross the Ethernet link. For the measurements in [2], only two pairs of timeout-buffer values are used (buffer of 10 packets with a 12 μs timeout and buffer of 100 packets with 120μs are used). The authors of [14] perform more extensive simulations on packet coalescing by using a timeout of 10 μs for testing 100 $Mbps$, 1000 $Mbps$ and 10 *Gbps* Ethernet links. Coordinated Transmission with EEE in 10 *Gbps* links is analyzed in [15]. Reviriego *et al.* show that for links with loads less than 50% this method can reduce the power consumption by powering down some PHY layer components and can allow longer cable lengths than the standard default 100 m. For LAN switches, a synchronized coalescing method is proposed by Mostowfi and Christensen [16], achieving potential power saving of about 40% for realistic TCP parameters (results were achieved via simulation). A new dynamic scheme is proposed in [17] but it lacks of full evaluation and comparison of the various parameters for both static and dynamic schemes. In all the above performance evaluation works, it is assumed that traffic is unidirectional and power saving is operated independently over the two link directions.

Our contribution. With our work, we add to the literature a model that considers the bidirectional behavior of 1 *Gbps* links and we evaluate its accuracy by means of EEE simulations based on real traffic traces. Additionally, we propose a complete performance evaluation of coalescing algorithms and show that static tuning of coalescing parameters can achieve near-optimal results in terms of power saving and packet delay.

3. EEE WITH GIGABIT ETHERNET

Energy Efficient Ethernet 802.03az [4] was standardized in September 2010. It aims to provide significant power saving in LANs. Formerly, the evolution of LANs led towards higher link speeds for faster communication and higher bandwidth, in order to satisfy the increased demand for data (link speeds from 10 $Mbps$ to 10 $Gbps$) without power consumption concerns. In fact, the electricity consumption of relatively "old" network interfaces remained in very low levels so the main concern of Ethernet component producers was not to save power. For example, in 100 $Mbps$ Ethernet links, the Ethernet devices consume about 200 mW of power [11]. However, higher speed Ethernet links (1 $Gbps$ or faster) require several Watts of power consumption [3] for each network interface. Considering a usual server that consumes around 200 W, a simple Ethernet device contributes to $\sim 10\%$ of this amount. Indeed, data centers and web hosting centers have a huge number of network interface cards which eventually generate a high cost (in terms of electricity bills). Thus, the idea of reducing the power consumption of Ethernet devices appears in the foreground.

Legacy Ethernet consumes a constant amount of power either with or without traffic, which makes it totally inefficient with typical Ethernet traffic profiles. This behavior results in a huge waste of power since it is well known that Ethernet links are inactive most of the time with utilization factors from 5% for a home PC to 30% for heavy loaded data servers [13, 16, 18]. EEE aims to reduce this waste of power and approach *power proportionality*, i.e., a power consumption proportional to the served traffic. The EEE standard introduces four new states for the Ethernet link, namely state "Active" (A) which corresponds to the busy period, state "Low Power Idle" (LPI) in which there is no traffic and the link consumes substantially less power than in state A ($\sim 90\%$ less power according to [12]), and states "Sleep" (S) and "WakeUp" (W) which correspond to the time spent during switching from state A to LPI and vice versa, respectively [5]. EEE specifications and state transition schemes are different for 100Base-T, 1000Base-T and 10GBase-T links. In particular, since we focus on commonly deployed 1 $Gbps$ links, in the following subsection we describe in detail how EEE 1000Base-T links behave, since they represent the only case in which the EEE standard accounts for the bidirectional behavior of traffic.

3.1 Gigabit EEE Link Operation

Behavior of 1Gbps EEE links. EEE 1000Base-T links can be in one of the following four states: Active (A), Sleep (S), WakeUp (W) and Low Power Idle (LPI). The state transition diagram is illustrated in Figure 1. Frame transmissions in either of the traffic directions only occur in state A. When the two network cards connected to the link complete transmitting all the buffered frames, the link enters state S as a transition to state LPI. If no frame arrives for T_s seconds while in state S, the link enters state LPI, during which power consumption is minimized. A frame arrival in state LPI results in the link transitioning to state W which lasts T_w seconds. After this wake interval, the link transitions to state A and any of the network interfaces connected to the link can transmit. Standard values for T_s and T_w are 182 μs and 16 μs, respectively. Thus, the fact that a frame arrival in the sleep interval (state S) causes an immediate transition to state A, avoids incurring in delays of up to 182

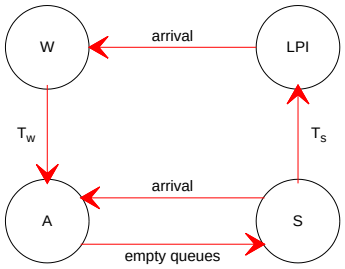


Figure 1: State transition diagram for EEE 1000Base-T.

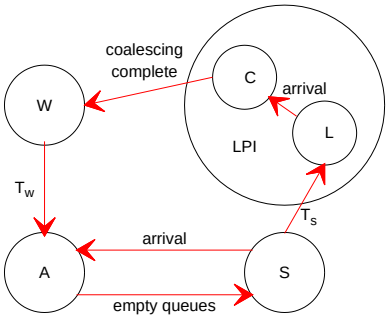


Figure 2: Modified state transition diagram for EEE 1000Base-T with coalescing.

μs, which can be quite large if compared to the transmission time of a single packet (e.g., 12 μs for a 1500-byte packet).

Behavior of 1Gbps EEE links with coalescing: When coalescing techniques are used, the transition from state LPI to state W is delayed. Therefore, we fictitiously split state LPI into two states, as shown in Figure 2: state L, which represents state LPI when there is no packet queued in the coalescing buffers—and which is equivalent to LPI in systems with no coalescing—and state C, in which coalescing buffers are not empty but neither the coalescing timer expired nor the coalescing buffers were completely full. Indeed, the system transitions from state C to state W as soon as one of the coalescing buffers gets full or the coalescing timeout expires. We remark that the newly introduced state C is fictitious, since the transition $L \rightarrow C$ does not represents a change of state for the EEE Ethernet hardware. State C simply represents the extension of an LPI interval due to coalescing operations, counting from the arrival of the first packet in one of the two coalescing queues. The time spent in state C, namely τ_c, is a random variable which depends on the size N_c of the coalescing buffers, and it is limited by the coalescing timeout T_c.

3.2 Efficiency Problems of EEE

As shown in previous works, such as [11, 14], the problem of EEE links (and especially in 1000Base-T links) is the transitioning time. When the traffic is scarce and packets are spaced rather that bursty, the EEE mechanism rarely allows the link to complete the transition to LPI. Thereby, the link spends more time transitioning than transmitting. Specifically, 1 $Gbps$ links require about 12 μs for transmitting a big packet, while for transitioning they spend at least $T_s+T_w=182+16=198$ μs. For smaller packets the analogy is even worst. For this reason, packet coalescing is proposed to avoid frequent state transitions and prolong the duration of state LPI. Packet coalescing tends to approach *energy proportionality* at the cost of additional delay to the packets. However, we will show in the next Section that the delay is negligible if compared to the energy benefits.

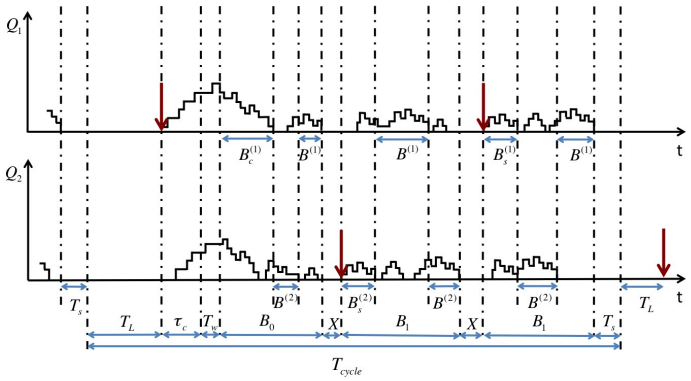


Figure 3: System cycle with coalescing.

4. A MODEL FOR BIDIRECTIONAL EEE GIGABIT LINKS

The goal of using EEE is to reduce the power consumption of Ethernet links; therefore, it is of great interest to evaluate the performance of EEE under different working conditions, and so models and simulators are needed to produce such a performance evaluation. We have developed an analytical model for EEE with bidirectional traffic with and without coalescing. We further modified the ns-3 simulator [8] to validate the model and evaluate the performance of EEE.

4.1 System Cycle and Power Saving

We model the behavior of the EEE link with coalescing as two $M/G/1$ queues, namely Q_1 and Q_2, in which the packet service rate is non-zero only when the EEE is in state A, where it equals a constant R corresponding to the link speed. Frames arrive to the transmission queues, representing the two network cards connected to the link, according to two independent Poisson processes with rates λ_1 and λ_2, respectively. We denote by $S_p^{(i)}$ the size of a single frame and by $E[S_p^{(i)}]$ the average frame size in link direction $i \in \{1, 2\}$.

Since events that cause state transitions can occur in either of the two link directions, for sake of tractability, we assume that these events are uncorrelated.

4.1.1 Cycle Analysis

The sample path of the queue can be viewed as a sequence of cycles as illustrated in Figure 3. A cycle starts with the packet arrival that induces the transition from state L to state C. Then, when the coalescing timer T_c expires or the number of coalesced packets reaches N_c, the link transitions to state W. Note that both transitions $L \to C$ and $C \to W$ can be caused by arrivals in either link directions. Note also that setting $T_c = 0$ and/or $N_c = 1$ yields the legacy EEE operation with no coalescing (i.e., the duration of state C is 0). In particular, transition $L \to C$ happens because of an arrival to queue Q_1 (namely, an arrival in direction 1) with probability $P_1 = \lambda_1/(\lambda_1 + \lambda_2)$, or because of an arrival to Q_2 (i.e., in direction 2) with probability $P_2 = 1 - P_1$.

The coalescing interval is followed by a busy period with the link in state A, whose duration is denoted by B_0. This initial busy period is followed by a random number ψ of sleep/active interval pairs, with each pair corresponding to an arrival in state S in either direction 1 or 2, before a time T_s has elapsed.

Note that the sleep time is then reduced to the random time interval between the start of state S and the next frame arrival, i.e., it is upper bounded by T_s. Finally, a sleep interval of duration T_s precedes the idle period T_L, whose random duration corresponds to the time interval before the beginning of a new cycle, i.e., before the next arrival in the system. We denote the length of a cycle by T_{cycle} and its average by $E[T_{cycle}]$.

Using results from renewal theory [19], we can focus on the system cycle, and compute the fraction of time spent in each link state as the ratio between the average time in each state in a cycle and the average cycle duration. We denote the average fraction of time spent in state α as η_α, for $\alpha \in \{A, C, L, S, W\}$. The following Theorem shows how to compute the average duration of a system cycle.

THEOREM 1. *For bidirectional EEE links in which arrivals in S are served immediately, the average cycle duration is given by:*

$$E[T_{cycle}] = (T_w + E[\tau_c]) \left[1 + \frac{1}{\lambda_1+\lambda_2}\left(\frac{\lambda_1\rho_1}{1-\rho_1} + \frac{\lambda_2\rho_2}{1-\rho_2}\right)\right] + \frac{e^{(\lambda_1+\lambda_2)T_s}}{\lambda_1+\lambda_2}\left[1 + \frac{\rho_1}{1-\rho_1} + \frac{\rho_1^2(2-\rho_1)(\lambda_1\rho_2+\lambda_2)}{2\lambda_1(1-\rho_1\rho_2)(1-\rho_1)^2} + \frac{\rho_2}{1-\rho_2} + \frac{\rho_2^2(2-\rho_2)(\lambda_2\rho_1+\lambda_1)}{2\lambda_2(1-\rho_1\rho_2)(1-\rho_2)^2}\right] \quad (1)$$

with $\rho_i = \lambda_i/\mu_i$, and $\mu_i = R/E[S_p^{(i)}]$, $i \in \{1, 2\}$.

PROOF. Consider the different intervals included in T_{cycle}, starting with the beginning of an L interval. The cycle is composed by the following elements: (i) an interval in state L, with random duration T_L until the first arrival to Q_1 or Q_2; (ii) a coalescing interval C of duration $\tau_c \le T_c$; (iii) a wake-up interval of fixed duration T_w; (iv) an interval B_0 lasting till the first epoch at which both queues Q_1 and Q_2 are empty; (v) an interval $X < T_s$ with exactly one arrival at time X, followed by an interval B_1 lasting until both queues are empty again; (vi) and finally a sleep interval of fixed duration T_s which triggers a new state L. Element (v) is optional, since it occurs only if there is one arrival within T_s seconds after B_0. Moreover, element (v) can repeat $\psi \ge 0$ times, until the idle interval following B_1 is longer than T_s. Each repetition of X and B_1 exhibits the same distribution because of the memoryless property of Poisson arrivals. However, busy intervals B_0 and B_1 can start either because of Q_1 or Q_2 activities, the two cases leading to different *conditional* average durations, as we will discuss in the following. Overall, the total cycle duration is:

$$T_{cycle} = T_L + \tau_c + T_w + B_0 + \psi(X + B_1) + T_s. \quad (2)$$

Note that $T_L + \tau_c$ is the time spent in state LPI during a cycle, while $B_0 + \psi B_1$ is the total time during which the link is active, and $T_s + \psi X$ is the time spent in state S. Let us now compute the average value for each of the elements composing the system cycle.

Interval T_L. State L lasts until the first packet arrival to Q_1 or Q_2. Since both arrival processes are Poisson, the first arrival behaves as the first of a Poisson flow with rate $\lambda_1 + \lambda_2$, i.e., with the following probability distribution:

$$f_{T_L}(t) = (\lambda_1 + \lambda_2)e^{-(\lambda_1+\lambda_2)t}, \quad t \ge 0; \quad (3)$$

and its average is then:

$$E[T_L] = \frac{1}{\lambda_1 + \lambda_2}. \quad (4)$$

Interval τ_c. For large values of N_c, we can approximate the duration of state C as the minimum time before $N_c - 1$ packet arrivals occur in either direction 1 or 2, and T_c. Let us denote with τ_{c_1} and τ_{c_2} the time before $N_c - 1$ arrivals appear in direction 1 or 2, respectively. Thereby, the time spent in coalescing is $\tau_c = \min\{\tau_{c_1}, \tau_{c_2}, T_c\}$. Recalling that the cumulative distribution function (CDF) of the minimum of n independent random variables is given by the following formula:

$$F_{\min\limits_{i=1\dots n}\{X_i\}}(x) = 1 - \prod_{i=1}^{n}(1 - F_{X_i}(x)), \tag{5}$$

and considering that the distributions of τ_{c_1}, τ_{c_2}, and T_c are as follows:

$$F_{\tau_{c_1}}(t) = \quad u(t)\left[1 - \sum_{k=0}^{N_c-2}\frac{(\lambda_1\, t)^k}{k!}e^{-\lambda_1\, t}\right]; \tag{6}$$

$$F_{\tau_{c_2}}(t) = \quad u(t)\left[1 - \sum_{k=0}^{N_c-2}\frac{(\lambda_2\, t)^k}{k!}e^{-\lambda_2\, t}\right]; \tag{7}$$

$$F_{T_c}(t) = \quad u(t - T_c); \tag{8}$$

where $u(t)$ is the unit step function, then the CDF of τ_c is given by:

$$F_{\tau_c}(t) = 1 - u(T_c - t)\left[\sum_{k=0}^{N_c-2}\frac{(\lambda_1 t)^k}{k!}e^{-\lambda_1 t}\right]\left[\sum_{k=0}^{N_c-2}\frac{(\lambda_2 t)^k}{k!}e^{-\lambda_2 t}\right], \tag{9}$$

where we used $1 - u(t) = u(-t)$. The average coalescing time is then as follows:

$$E[\tau_c] = \int_0^{T_c} t \cdot dF_{\tau_c}(t). \tag{10}$$

However, assuming that we can tune N_c and T_c in a way that the coalescing queues do not fill completely with probability almost one, then $E[\tau_c] \simeq T_c$. Note that the latter assumption is realistic for actual implementations since the power saving increases with N_c, although delay increases too and we want to bound it to T_c.

Interval T_w. The wake-up interval which precedes the first busy interval has fixed duration T_w.

Interval B_0. This interval is composed of various subparts, as shown in Figure 3. After the link transitions to state A from state C, at least one queue is not empty. The system remains busy until *both* queues are empty again. Let us observe the system from the viewpoint of the queue that received the packet that caused the beginning of the the coalescing state (transition $L \to C$). We denote with $B_c^{(i)}$, $i \in \{1, 2\}$, the first busy period seen at queue i only, after the transition $C \to A$. This is the busy period of an $M/G/1$ queue, for which the average depends on the arrival rate λ_i, the mean service time $E[S_p^{(i)}]/R$, and the queue size $Z_c^{(i)}$ at the beginning of the busy period [19]:

$$E[B_c^{(i)}] = \frac{E\left[Z_c^{(i)}\right]\, E\left[S_p^{(i)}\right]/R}{1 - \rho_i} = \frac{E\left[Z_c^{(i)}\right]\, \rho_i}{\lambda_i(1 - \rho_i)}. \tag{11}$$

If queue i is the one who received the packet that triggered the transition $L \to C$, then $E\left[Z_c^{(i)}\right] = 1 + \lambda_i\,(T_w + E[\tau_c])$, i.e., the initial queue size equals the arrival that triggers the coalescing timer, plus the average number of Poisson arrivals during the average coalescing time $E[\tau_c]$ and the wake-up interval T_w. The probability that queue Q_1 is the one who initiates the coalescing procedure is simply given by the probability of having a Poisson arrival with rate λ_1 before a Poisson arrival with rate λ_2, counting from the beginning of state L. I.e.:

$$Pr(Q_1 \text{ triggers coalescing}) = \quad \frac{\lambda_1}{\lambda_1 + \lambda_2}; \tag{12}$$

$$Pr(Q_2 \text{ triggers coalescing}) = \quad \frac{\lambda_2}{\lambda_1 + \lambda_2}. \tag{13}$$

With no loss of generality, assume now that queue Q_1 triggers the coalescing. Therefore, as observed from Q_1, the system goes through a busy period $B_c^{(1)}$, at the end of which Q_1 is empty with probability 1, while Q_2 is empty with probability $1 - \rho_2$. If Q_2 is not empty, let us observe the followup in the evolution of the system from the viewpoint of Q_2: there is a busy period $B^{(2)}$ for Q_2, at the end of which Q_2 will be empty, while Q_1 can be empty with probability $1 - \rho_1$. The process can replicate by alternating busy intervals $B^{(1)}$ and $B^{(2)}$, i.e., we alternate the observation of the system from the viewpoint of a queue or the other, until the observation of the queue status at the end of a busy period reveals that both queues are empty. At that point we have a transition $A \to S$. Busy periods $B^{(i)}$ have different average duration with respect to $B_c^{(i)}$. In fact, the initial backlog of the queue in $B^{(i)}$ is not $Z_c^{(i)}$. Using the Pollaczek-Khinchin mean formula to estimate the average backlog of an $M/G/1$ queue at a random observation point, we would have $\lambda_i\, E[S_p]/R + \frac{\lambda_i^2\, E\left[S_p^2\right]/R^2}{2(1-\rho_i)}$ as initial backlog. However, we are interested in the conditional initial backlog $Z^{(i)}$, given that the observed queue is not empty (otherwise there is no busy period), which happens with probability ρ_i. Therefore we have $E\left[Z^{(i)}\right] = 1 + \frac{\rho_i}{2(1-\rho_i)}$, and the observed busy periods are given by:

$$E\left[B^{(i)}\right] = \rho_i \frac{2 - \rho_i}{2\lambda_i(1 - \rho_i)^2}. \tag{14}$$

Following the above procedure, one can compute the average duration of the first period after state C during which either queue 1 or 2 are busy:

$$E[B_0] = \quad \frac{1}{\lambda_1 + \lambda_2}\left[\lambda_1 E\left[B_c^{(1)}\right] + \lambda_2 E\left[B_c^{(2)}\right] + \frac{\rho_1(\lambda_1\rho_2 + \lambda_2)E\left[B^{(1)}\right]}{1 - \rho_1\rho_2} + \frac{\rho_2(\lambda_2\rho_1 + \lambda_1)E\left[B^{(2)}\right]}{1 - \rho_1\rho_2}\right]. \tag{15}$$

Interval X. The interval X between the end of a busy period and the beginning of the next busy period is exponentially distributed with rate $\lambda_1 + \lambda_2$, given that the next arrival occurs within T_s seconds:

$$f_X(t) = \frac{(\lambda_1 + \lambda_2)e^{-(\lambda_1+\lambda_2)t}}{1 - e^{-(\lambda_1+\lambda_2)T_s}}, \quad t \in [0, T_s].$$

Accordingly, the average of X is as follows:

$$E[X] = \frac{1}{\lambda_1 + \lambda_2} - \frac{T_s}{e^{(\lambda_1+\lambda_2)\,T_s} - 1}.$$

Interval B_1. Similarly to the case of B_0, this interval is composed by various subparts. The first part is $B_s^{(i)}$, which is the first busy interval on either Q_1 or Q_2 after the period X. Since packets are served immediately when they arrive

in state S, the initial backlog is exactly 1. For the rest, the computation of $B_s^{(i)}$ is analogue to the one of $B_c^{(i)}$:

$$E[B_s^{(i)}] = \frac{\rho_i}{\lambda_i(1-\rho_i)}.$$

The following alternating busy intervals are exactly like for the case of B_0, i.e., we have intervals $B^{(1)}$ and $B^{(2)}$.

Considering that each interval B_1 starts because of an arrival in Q_1 or Q_2 before T_s expires, and since arrivals for Q_1 and Q_2 in state S are independent and both follow a Poisson process, the probability of starting an interval B_1 due to arrivals for Q_i is $\frac{\lambda_i}{\lambda_1+\lambda_2}$.

Putting together the pieces, the average of B_1 is as follows:

$$E[B_1]= \frac{1}{\lambda_1+\lambda_2}\left[\lambda_1 E\left[B_s^{(1)}\right]+\lambda_2 E\left[B_s^{(2)}\right] + \frac{\rho_1(\lambda_1\rho_2+\lambda_2)E\left[B^{(1)}\right]}{1-\rho_1\rho_2}+\frac{\rho_2(\lambda_2\rho_1+\lambda_1)E\left[B^{(2)}\right]}{1-\rho_1\rho_2}\right]. \quad (16)$$

Number of repetitions ψ. Busy intervals B_1 occur if the residual interarrival time at the end of the previous busy interval is shorter than T_s. Since arrivals are Poisson, the probability of having no arrivals in any link direction in T_s is $P_0 = e^{-(\lambda_1+\lambda_2)\,T_s}$. Thereby, the number $\psi \geq 0$ of busy periods of type B_1 in a cycle, i.e., not counting B_0, can be seen as the number of consecutive successes of a geometric random variable ψ with success probability $1-P_0$. Hence, its average value is:

$$E[\psi] = \frac{1-P_0}{P_0} = e^{(\lambda_1+\lambda_2)\,T_s} - 1.$$

Interval T_s. The sleep interval which follows the last busy interval before entering state L has fixed duration T_s.

Average cycle duration. Putting together the results obtained for the cycle components, after some algebraic elaboration, result (1) follows.

□

COROLLARY 1. *The fraction of time spent in LPI is:*

$$\eta_{LPI} = \frac{\frac{1}{\lambda_1+\lambda_2}+E[\tau_c]}{E[T_{cycle}]}. \quad (17)$$

PROOF. The time spent in LPI corresponds to states L and C, i.e., intervals T_L and τ_c, as described in the proof of Theorem 1; therefore the proof follows. □

COROLLARY 2. *The fraction of time spent in WakeUp state is given by:*

$$\eta_W = \frac{T_w}{E[T_{cycle}]}. \quad (18)$$

PROOF. The time spent in $WakeUp$ state is constant in each cycle and corresponds to the interval T_w. Therefore the proof follows. □

COROLLARY 3. *The fraction of time spent in Sleep state is given by:*

$$\eta_S = \frac{E[X]E[\psi]+T_s}{E[T_{cycle}]}. \quad (19)$$

PROOF. The time spent in *Sleep* state corresponds to intervals X, which are repeated ψ times, plus a complete sleep interval of T_s seconds, occurring once in a cycle, just before entering state L. Considering the proof of Theorem 1, the proof follows. □

COROLLARY 4. *The fraction of time spent in Active state is given by:*

$$\eta_A = \frac{E[B_0]+E[B_1]E[\psi]}{E[T_{cycle}]}. \quad (20)$$

PROOF. The time spent in *Active* state is given by the sum of busy intervals during which at least one network interface transmits. Therefore, considering the proof of Theorem 1, the proof follows. □

4.1.2 Power Saving

Using the results of the analysis carried out for the cycle duration, we can now compute the power saving factor Φ achieved by EEE with or without coalescing.

THEOREM 2. *The average power consumption achieved by EEE is proportional to the fraction of time spent in LPI:*

$$\Phi \propto \eta_{LPI}. \quad (21)$$

PROOF. The average power consumption over a system cycle, is computed by considering that the power consumption in states W (namely $P^{(W)}$) and S (namely $P^{(S)}$) is practically the same as in state A (namely $P^{(A)}$), while in LPI (i.e., $P^{(LPI)}$ in states L and C), the power consumption decreases by a factor $k \simeq 10$, as experimentally shown by Reviriego *et al.* [12]. In legacy gigabit cards, the power consumption $P^{(legacy)}$ is practically constant and equals the one consumed in state A in an EEE card. Therefore we have the following expression for the power saving factor Φ:

$$\Phi = 1 - \frac{\sum_{\alpha\in\{A,S,LPI,W\}}\eta_\alpha P^{(\alpha)}}{P^{(legacy)}} \simeq \frac{k-1}{k}\cdot\eta_{LPI}. \quad (22)$$

Φ is thus shown to be proportional to η_{LPI} with a proportionality factor $(k-1)/k \simeq 0.9$. □

The power saving achieved with EEE is proportional to η_{LPI} and the values of Φ and η_{LPI} are similar. As a consequence, in the rest of the paper we will refer to *power saving performance* either in case of actual power saving figures or when discussing η_{LPI} values.

4.2 Packet Delay

As concerns the average frame delay we differentiate between packet arrivals in the different states A, S, L, C, W.

Delay D_A of packets arriving in A. If a packet arrives in state A, we can use results for M/G/1 queues. Therefore, the average waiting time can be simply computed by means of the P-K formula [19]:

$$D_A^{(i)} = \frac{\lambda_i E[\sigma_i^2]}{2(1-\rho_i)}, \quad i \in \{1,2\}, \quad (23)$$

where $\sigma_i = S_p^{(i)}/R$ is the random service time, with $E[\sigma_i] = 1/\mu_i$. The statistics of σ_i depends on the packet size distribution. For the sake of simplicity, here we assume that packet size is constant and equal to $1/\mu_i$, so that we use $E[\sigma_i^2] = 1/\mu_i^2$, which yields $D_A^{(i)} = \frac{\rho_i}{2\mu_i(1-\rho_i)}$.

Note that the resulting delay is different in the two link directions.

Delay D_S of packets arriving in S. If a packet arrives while the device is in state S, the device will directly transition to state A and the packet is immediately served. Thus, the delay is $D_S = 0$.

Delay D_L of packets arriving in L. Only one packet can arrive while the device is in state L, which triggers an immediate transmission to state C. The delay of this packet is at most $T_c + T_w$, in case of scarce traffic which yields the expiration of the coalescing timeout. More in general, the average queuing delay experienced by this packet is the sum of the average coalescing time, given in Eq. (10), plus the constant wake-up time T_w:

$$D_L = E[\tau_c] + T_w. \tag{24}$$

Delay D_C of packets arriving in C. When a packet arrives in state C, it suffers from (i) the residual coalescing interval, (ii) the constant wake-up interval T_w, and (iii) the time to process and transmit packets already present in the queue at the arrival epoch of the new packet.

Considering that Poisson arrivals are uniformly distributed over time, we estimate the average residual coalescing time as $E[\tau_c]/2$. Correspondingly, the average queue size at the arrival epoch is $\lambda_i E[\tau_c] + \frac{\lambda_i}{\lambda_1+\lambda_2}$ for an arrival in direction $i \in \{1,2\}$, where the second term represents the probability that the packet triggering the $L \to C$ transition belongs to direction i. The average cumulative serving time for those packets is then $\rho_i \left(\frac{E[\tau_c]}{2} + \frac{1}{\lambda_1+\lambda_2} \right)$. Therefore, the delay suffered by a packet arriving in state C is, on average:

$$D_C^{(i)} = T_w + \frac{E[\tau_c]}{2} + \rho_i \left(\frac{E[\tau_c]}{2} + \frac{1}{\lambda_1+\lambda_2} \right),\ i \in \{1,2\}. \tag{25}$$

Considering that the delay D_C is affected by the arrival rate, this delay assumes different average values for packets sent in the two different link directions.

Delay D_W of packets arriving in W. If a packet arrives while the device is in state W, the delay is composed of (i) the average residual wake-up time (for uniformly distributed Poisson arrivals, this equals $T_w/2$), and (ii) the required time to serve all packets arrived earlier, since the beginning of state C: $\frac{\lambda_i}{\lambda_1+\lambda_2} + \lambda_i(E[\tau_c] + T_w/2)$ packets, on average (again the first term is due to the packet which triggers the $L \to C$ transition). Therefore, the delay is:

$$D_W^{(i)} = \frac{T_w}{2} + \rho_i \left(\frac{1}{\lambda_1 + \lambda_2} + E[\tau_c] + T_w/2 \right),\ i \in \{1,2\}. \tag{26}$$

This delay is different for different traffic directions, as it was the case for D_A and D_C.

Average delay of a packet. To find the average delay that a packet can suffer, we need to compute the probability that a packet arrives in any of the possible EEE states. For each link direction, these probabilities can be seen as the average number of packets received in each of the different states divided by the average number of arrivals in a system cycle.

There is only one packet per cycle arriving in state L: it belongs to link direction i with probability $\lambda_i/(\lambda_1 + \lambda_2)$, which is then the average number of packets received in state L in direction i, namely $n_L^{(i)}$. Similarly, since there are ψ arrivals per cycle in state S, the average number of packet arrivals in direction i in state S is given by $n_S^{(i)} = \frac{\lambda_i}{\lambda_1+\lambda_2} E[\psi]$.

Since arrivals follow a Poisson process, packets received in link direction i during the fixed-length wake-up interval, which is present only once in a cycle, are $n_W = \lambda_i T_w$, on average. Similarly, the number of arrivals in direction i during the coalescing interval (state C) is $n_C = \lambda_i E[\tau_c]$, on average. Eventually, arrivals in direction i in state A are the total number of (Poisson) arrivals in a cycle less the arrivals in the other states, i.e., $n_A^{(i)} = \lambda_i E[T_{cycle}] - n_S^{(i)} - n_W^{(i)} - n_L^{(i)} - n_C^{(i)}$. As a result, the corresponding average delay that a packet suffers in direction i is as follows:

$$D_i = \frac{\sum_{\alpha \in \{A,S,L,C,W\}} n_\alpha^{(i)} D_\alpha^{(i)}}{\lambda_i E[T_{cycle}]}, \quad i \in \{1,2\}. \tag{27}$$

5. DYNAMIC COALESCING STRATEGIES

In this section we present two classes of algorithms that can be used to dynamically match the coalescing parameters to the traffic conditions. These algorithms will allow us to explore the performance of dynamic packet coalescing, which has not been addressed so far in the literature.

The rationale behind proposing dynamic coalescing is that static configurations might incur in high delays when the traffic is low. For instance, to increase power saving, coalescing techniques try to avoid short and frequent transmission bursts by means of large T_c and N_c values. However, as soon as the traffic intensity causes frequent coalescing timeouts, the latency often exceeds T_c. Therefore, in a static configuration, one cannot use very large values of T_c when the traffic intensity can be low with non-negligible probability. Similarly, large values of N_c increase the achievable power saving, but cause high latency due to queue backlog to be served after the coalescing period. Using a static configuration for all traffic conditions might incur in the following problem: when the traffic is low the coalescing timer expires frequently, while when the traffic is high one or both coalescing buffers get full too quickly. In the former case, the experienced delay might be too high, while in the latter case, the achieved power saving might be far from optimal.

In the following subsections we present two classes of dynamic coalescing algorithms for EEE and evaluate their behavior in terms of power saving and packet delay. The first algorithm, Dynamic Timeout Algorithm, makes use of tunable coalescing timeout of duration T_c, while the second one, Dynamic Queue Size Algorithm, uses tunable coalescing buffers of variable size N_c.

5.1 Dynamic Timeout

In this class of algorithms, as indicated by its name, the adjustable parameter is the coalescing timeout T_c, while the coalescing buffers have fixed size N_c. We recall that T_c is defined as the maximum interval that EEE network cards can remain in state C after the arrival of the first packet in state L, thus extending the normal EEE power saving interval. As stated before, when the coalescing timeout expires, both Ethernet interfaces start transmitting the queued packets to the other link edge. The goal of this first class of algorithms is to keep the system in state C for as long as it is needed to fill up at least one coalescing buffer, i.e., to adjust T_c so that the coalescing operation (i.e., the duration of state C) lasts $\sim T_c$ seconds and the maximum coalescing gain is achieved by filling up the coalescing buffer.

The algorithm's behavior is described by means of pseudocode in Algorithm 1. In this algorithm, the value of T_c keeps changing when the transition $C \to A$ occurs. If the transition occurs because of a timeout expiration, then T_c is

incremented, unless it reaches a maximum value $T_c^{\max}$. Its maximum value may depend on various factors but the most important is the maximum delay tolerance that is allowed either by applications, or by quality of service constraints. Similarly, if the transition $C \rightarrow A$ occurs because one of the coalescing buffers gets full, then T_c is decremented, unless it reaches its minimum allowable value $T_c^{\min}$. Therefore, the algorithm tries to adjust T_c in a way that state C lasts approximately T_c seconds while trying both to avoid unnecessary long timeouts and to fill up coalescing buffers.

In Algorithm 1, increments and decrements of T_c can follow different strategies. In particular, we consider that both increments and decrements can be either additive or multiplicative. Therefore, to fully specify the behavior of Algorithm 1, we need to specify (i) whether additive or multiplicative increments and decrements are used, (ii) the steps used in case of additive operation (δ_{up} for increments and/or δ_{down} for decrements), or the multiplicative factors used in case of multiplicative operation (γ_{up} for increments and/or γ_{down} for decrements), and (iii) the range $[T_c^{\min}, T_c^{\max}]$ in which T_c can be adjusted.

The performance of Algorithm 1 depends on the target N_c value fixed in the system. Throughout our experiments, we use $T_c^{\min} = 0.1\ ms$ and $T_c^{\max} = 100\ ms$, while we tested all combinations of additive and multiplicative increments and decrements, with various values for δ_{up}, δ_{down}, γ_{up}, and γ_{down}.

5.2 Dynamic Queue Size

The second algorithm is similar to the first, but it adjusts the coalescing buffer size N_c instead of the coalescing timeout T_c. In Algorithm 2, N_c is dynamically and automatically tuned in order to adapt the coalescing operation to achieve

```
Data: δup or γup, δdown or γdown, Tc^min, Tc^max, Nc
if Transition C → A then
    if Coalescing timeout expired then
        if Tc < Tc^max then
            increase Tc;
        end
    else
        if Tc > Tc^min then
            decrease Tc;
        end
    end
    restart the coalescing timeout with new Tc;
end
```

Algorithm 1: Dynamic Timeout Algorithm.

```
Data: δup or γup, δdown or γdown, Nc^min, Nc^max, Tc
if Transition C → A then
    if Coalescing timeout expired then
        if Nc > Nc^min then
            decrease Nc;
        end
    else
        if Nc < Nc^max then
            increase Nc;
        end
    end
    restart the Tc;
end
```

Algorithm 2: Dynamic Queue Algorithm.

a target coalescing delay T_c. I.e., the target of Algorithm 2 is to keep the system in state C for about T_c seconds and during this interval, accumulate as many packets as possible.

In Algorithm 2, when the traffic intensity is such low that the coalescing timeout expires before N_c packets are queued in any of the two coalescing buffers, the algorithm decreases N_c. The minimum value for N_c is $N_c^{min} = 2$, since smaller values would not result in any coalescing operation. Conversely, when the traffic increases and at least one coalescing buffer fills up before the coalescing timeout expires, the algorithm increases N_c up to its maximum value N_c^{max}. Similarly to what stated for Algorithm 1, increments and decrements can be either additive or multiplicative, with parameters that we keep calling δ_{up}, δ_{down}, γ_{up}, and γ_{down} like in the previous algorithm.

The performance of Algorithm 2 depends on the target value of T_c fixed in the system. Throughout our experiments, we use $N_c^{\min} = 2\ pkt$ and $N_c^{\max} = 10000\ pkt$, while we tested all combinations of additive and multiplicative increments and decrements, with various values for δ_{up}, δ_{down}, γ_{up}, and γ_{down}.

In the next Section we compare the two classes of dynamic coalescing algorithms with static algorithms.

6. PERFORMANCE EVALUATION

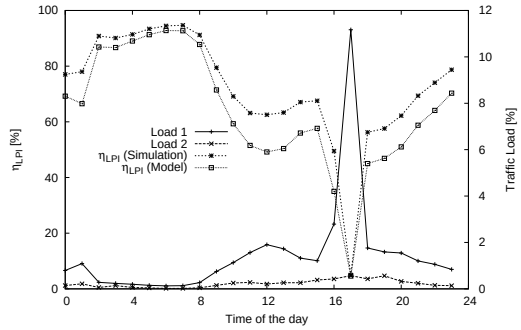
In this section we use ns-3 simulations and the model described in Section 4 to assess the performance of coalescing techniques over 1 *Gbps* EEE links. Note that the model cannot be used for dynamic coalescing, so that we will use simulations for the performance assessment of the *Dynamic Timeout* and the *Dynamic Queue Size* algorithms. However, we will show that static coalescing performs almost as well as dynamic coalescing, which reveals that our model can be used to quickly predict potential EEE performance with and without dynamic coalescing.

We investigate on the power saving and on the delay of the packets by using different δ and γ parameters for the dynamic algorithms presented in Section 5, and for various values of N_c and T_c for both static and dynamic approaches. Notably, for our evaluation we use real traffic traces that we captured at two of the firewall interfaces of InterHost, a large web hosting center in Madrid. Therefore, our results correspond to realistic traffic patterns, as well as realistic power saving and packet delay figures achieved by means of EEE and coalescing techniques.

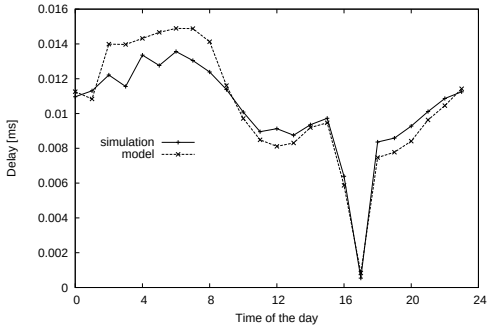
To run simulations we modified the ns-3 simulator to implement EEE and packet coalescing. First, we designed and coded a novel Ethernet channel object in ns-3. Such an Ethernet channel can simulate the bidirectional behavior of Ethernet links. Second, we added on the net devices the EEE functionality, i.e., we defined the EEE states. Third, we implemented packet coalescing and coordinated packet transmission so that a simulated EEE link enters state L only when both traffic directions are inactive for T_s seconds, and exits state C only when coalescing operations are complete either because the coalescing timer expires or one of the coalescing buffers fills up.

6.1 Model vs. Simulation

Here we show that model and simulation return similar results, thus validating the model proposed in Section 4. High precision timestamped real bidirectional traces have been collected and used as input to the simulator. High precision



(a) Load and power saving opportunities (η_{LPI}).



(b) Delay in the most delayed link direction.

Figure 4: EEE model and simulation results with real traces, sampling a one-day traffic pattern (without coalescing).

timestamping is very important for such kind of simulations as we have shown in [13]. As concerns the model, we used as input a few simple but significant statistical parameters, i.e., the average load, arrival rate and packet size in the two link directions.

6.1.1 Without Coalescing

We first consider a plain EEE scenario, with no coalescing. In Figure 4, we depict η_{LPI} and the packet delay computed via our modified ns-3 simulator and via the analytical model that we presented in Section 4.

Figure 4a illustrates the EEE power saving (in terms of η_{LPI}) for a typical day, and the measured traffic sampled once every 60 minutes. The load is very low most of the time, i.e., less than 1%, but we observed an exceptional peak value of medium traffic ($\sim$10%) at 5pm. This traffic behavior is in line with typical data center loads as described in [20].

We can observe that the model we previously presented estimates the power saving with a good accuracy in either low (0–1%) or medium ($\sim$10%) traffic conditions. Moreover, in Figure 4a, EEE can save most of the time more than 50% of power. In extreme cases (but not infrequently, since they constitute about 1/3 of the day's samples) the time spent in power saving (LPI) exceeds 90%. Remarkably, EEE enables substantial power saving ($\eta_{LPI} > 90\%$) when the traffic is limited to few percents of the link capacity. However, EEE does not appear to be able to save much power when the traffic surges to relatively low peaks, e.g., to 10% of the link capacity. With higher loads, we can safely assume that a plain EEE approach would bring negligible power saving. This justifies the research for EEE enhancements that would allow significant savings even when traffic reaches typical utilization levels such as 10% of the link capacity.

The accuracy of our model, in terms of delay performance, is evaluated through Figure 4b over the same traffic traces studied in Figure 4a. The Figure reports the delay suffered in the link direction over which the delay due to EEE state transitions is higher. Results achieved with model and simulation differs by at most $3\mu s$, which is a negligible quantity. The Figure also shows that packet delay is higher when the traffic offered to the link is lower. Moreover, delays due to EEE are never relevant, since they are comparable to or shorter than the duration a single packet transmission time.

In general, we have shown that our model can be safely used to estimate EEE performances in terms of both power saving and delay when no coalescing is adopted. In the next subsection we will show the correctness of our model also for the case of EEE with static coalescing. Before that, in light of our traffic measurements and power saving estimates, we enlighten the potential economical benefit of EEE.

Economical relevance of EEE power saving. From a purely economical point of view, our results for EEE are considerable. For instance, let us consider a large data center, e.g., the one of Google [21]. The data center includes over $40,000$ servers, and each server has about 3 connected network ports, on average [22]. Each port may consume about 2 Watts using legacy NICs [3]. A typical load distribution consist in having about 40% of the link at almost zero load, 48% at about 10% load, and the rest of the links operated at higher load [20]. Thus, considering our results and that the average cost of electricity in Europe is about 0.15 $Euros$ per KWh, we can roughly estimate that EEE would reduce the annual electricity bill by 135,000 Euros. This is a rough estimate, and it does not include additional savings due to the reduced need for air conditioning.

Our preliminary results are particularly interesting since: (i) we can expect (and we will actually show it in the following) that packet coalescing can further boost the power saving achievable by means of EEE; and (ii) faster Ethernet cards, i.e., 10 *and* 100 *Gbps*, consume even more power (at least two and five times more, respectively), so that the potential for power saving is greater for higher data rates.

6.1.2 With Static Coalescing

To validate our model for EEE with static packet coalescing, in Table 1 we list some representative results achieved with model and simulation. We use the traffic traces collected at two different links in InterHost, Madrid (Spain). One of the links had low traffic ($< 10\%$) and the second one had traffic peaks of more than 30%. For each trace we report loads, arrival rates and average packet sizes for each link direction. We simulate the static coalescing algorithm for multiple combinations of N_c and T_c, and report the achieved results for the fraction of time spent in LPI (η_{LPI}), and for the average packet delay in the two directions (D_1 and D_2). We also use the average traffic descriptors reported in the Table to evaluate η_{LPI}, D_1 and D_2 through our model.

We observe that we have high traffic in one direction and low traffic in the other direction in all traces but the last, where the traffic is almost balanced. We note that high loads correspond to large packet sizes, which, in turn, correspond to the typical behavior of TCP traffic, i.e., large packets in one direction and small acknowledgements in the other direction. From Table 1, we can also observe that the model approximates very well both the time spent in LPI, η_{LPI}, and the average delay in the two link directions. η_{LPI} is

Table 1: η_{LPI} and average delays computed with the model and via simulation

ρ_1 [%]	ρ_2 [%]	$S_p^{(1)}$ [bytes]	$S_p^{(2)}$ [bytes]	λ_1 [pkts/s]	λ_2 [pkts/s]	T_c [ms]	N_c [pkts]	η_{LPI} [%]		D_1 [ms]		D_2 [ms]	
								simul.	model	simul.	model	simul.	model
0.20	0.06	802	281	310	268	5	50	97.45	96.84	3.157	3.067	3.383	3.063
						5	100	97.45	96.84	3.157	3.067	3.383	3.062
						10	50	98.29	98.11	5.925	5.657	6.140	5.652
						10	100	98.30	98.11	5.942	5.660	6.147	5.652
						20	100	98.92	98.91	11.130	10.725	11.306	10.710
0.71	39.69	67	1512	13187	32815	5	50	0.001	0.001	0.002	0.006	0.002	0.011
						5	100	1.69	1.74	0.001	0.026	0.001	0.039
						10	50	1.72	0.88	0.003	0.006	0.004	0.011
						10	100	1.79	1.74	0.005	0.027	0.005	0.039
						20	100	1.85	1.75	0.007	0.027	0.009	0.039
0.87	29.64	83	1477	13048	25084	5	50	7.87	4.56	0.024	0.046	0.019	0.060
						5	100	7.97	8.64	0.028	0.174	0.023	0.225
						10	50	7.93	4.57	0.041	0.046	0.029	0.060
						10	100	8.71	8.65	0.068	0.174	0.058	0.225
						20	100	9.20	8.66	0.101	0.175	0.082	0.225
0.98	53.66	69	1500	17769	44719	5	50	0.10	0.03	0.000	0.000	0.001	0.004
						5	100	0.14	0.06	0.001	0.000	0.003	0.005
						10	50	0.10	0.03	0.000	0.000	0.001	0.004
						10	100	0.14	0.06	0.001	0.000	0.003	0.005
						20	100	0.14	0.06	0.001	0.000	0.003	0.005
3.72	0.37	1180	165	3938	2778	5	50	85.03	90.90	1.896	2.442	2.123	2.363
						5	100	85.29	90.90	1.954	2.442	2.153	2.364
						10	50	88.26	94.08	3.971	4.946	3.819	4.786
						10	100	90.10	94.10	4.331	4.972	4.453	4.811
						20	100	91.90	95.82	8.926	9.031	8.256	9.707
5.06	0.50	1170	165	5408	3809	5	50	78.76	88.10	1.727	2.380	1.869	2.277
						5	100	79.21	88.10	1.815	2.380	1.922	2.277
						10	50	82.05	91.63	3.854	4.334	3.160	4.146
						10	100	85.52	92.18	4.079	4.914	4.109	4.701
						20	100	87.54	94.17	7.700	8.029	6.910	8.642
9.74	6.28	1165	855	10452	9182	5	50	65.61	64.14	1.390	1.601	1.252	1.550
						5	100	70.55	66.28	1.947	1.850	1.685	1.792
						10	50	69.70	64.43	1.593	1.632	1.634	1.581
						10	100	79.90	75.90	3.610	3.879	3.663	3.757
						20	100	79.84	76.05	3.710	3.928	3.670	3.805

estimated with ±5% deviation in most of the cases (90%), while average delay estimations are subject to an error which is of the order of 10% for high delay cases (several ms), and a few μs for the case of low delays (tens of μs or less).

Overall, under any of the tested traffic conditions and coalescing configurations, the results achieved through the model are accurate enough with respect to simulations. However, running simulations requires much more time and computational resources than the model. Hence, our model results in a suitable tool for the quick evaluation of (i) EEE potentials and (ii) coalescing configuration effectiveness, under any traffic condition.

6.2 Static vs. Dynamic Coalescing

We now compare the performance of static and dynamic coalescing. We use simulation only, since our model was not designed to predict the behavior of dynamic coalescing schemes. However, we will show that static and dynamic coalescing achieve very similar results, so that developing a model for EEE with dynamic coalescing is unnecessary.

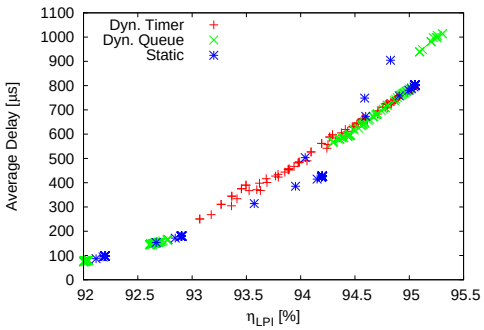
6.2.1 Static Coalescing

Table 1 reveals that coalescing enables high power saving opportunities in a variety of load conditions. In particular, even in presence of loads of the order of 10%, power can be saved 60 to 80% of the time. High value of N_c and T_c would even allow for relevant power saving ($\sim 10\%$) under high traffic ($\sim 30\%$). The Table also reveals that delays grow fast with N_c and T_c. However, under high load, the achieved average delay assumes values not greater than a few hundreds of μs. High delays can be suffered only when the traffic is low and the coalescing parameters are high.

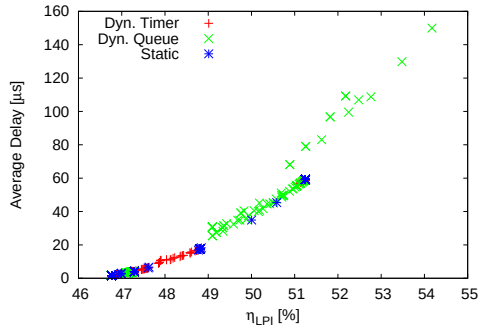
Notably, using $N_c \in [50, 100]$ packets and $T_c = 10\,ms$ yields high power saving under all reported cases, with quite limited and acceptable delays. Hence, static coalescing appears to be near-optimal under a wide range of traffic conditions.

In addition to what reported in Table 1, we tested many possible combinations for many traffic traces, with N_c ranging from 2 to 10000 packets, and T_c ranging from 0.1 to 100 ms. Due to space limitations, we omit here a complete description of our results with static coalescing and—since configurations resulting in high delay are undesirable—we limit our discussion only to cases with average delay below 1 ms. Specifically, we selected four different traffic traces, corresponding to the most typical traffic loads. Figure 5 reports the results for both static and dynamic algorithms under the following traffic conditions; Figure 5a is for $\rho_1 = 0.2\%$ and $\rho_2 = 0.06\%$, Figure 5b for $\rho_1 = 5.1\%$ and $\rho_2 = 0.1\%$, Figure 5c for $\rho_1 = 10.9\%$ and $\rho_2 = 0.2\%$ and Figure 5d for $\rho_1 = 39.7\%$ and $\rho_2 = 0.7\%$. Since we are interested in the potential performance of coalescing in terms of power saving and delay, Figure 5 plots the values for the delay in the most loaded link direction as a function of the achieved η_{LPI}. Each blue "$*$" marker in the Figure is obtained under a different coalescing configuration for N_c and T_c.

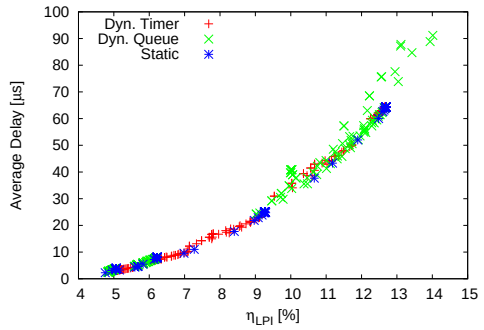
Figures 5a to 5d show that higher power saving corresponds to higher delay. However, delay can be minimized in exchange of small power saving reduction. Under low loads, as in Figure 5a, the value of η_{LPI} stays well above 90% in all cases, which means that coalescing of a very few packets (e.g., $N_c = 2$), combined with a short coalescing timeout (e.g., $T_c = 1\ ms$), is more than enough to achieve high power saving with very limited packet delay. Conversely, under high load (e.g., Figure 5d), there is no static configuration that can bring high power saving with bounded delay.



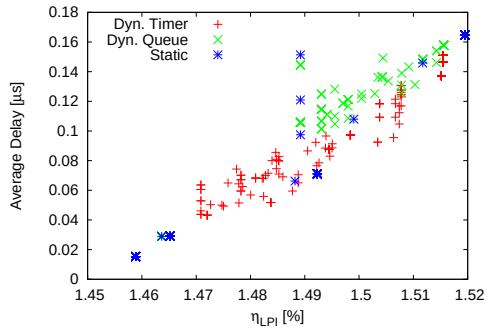
(a) $\rho_1 = 0.2\%$, $\rho_2 = 0.06\%$.



(b) $\rho_1 = 5.1\%$, $\rho_2 = 0.1\%$.



(c) $\rho_1 = 10.9\%$, $\rho_2 = 0.2\%$.



(d) $\rho_1 = 39.7\%$, $\rho_2 = 0.7\%$.

Figure 5: η_{LPI} vs. delay performance using different coalescing algorithms under various traffic conditions.

6.2.2 *Dynamic Timeout*

The Dynamic Timeout Algorithm "helps" the EEE coalescer to adapt its T_c value to the traffic conditions. In Figure 5, in addition to results for static coalescing, we plot the average delay (the maximum in the two link directions) versus η_{LPI} for $N_c \in [2, 10000]$, δ from 0.1 to 10 ms, and γ from 1 to 100%. Adopted traffic traces are the same as for the evaluation of static coalescing. Similarly to the static coalescing case, we report results for the cases in which the average added delay per packet is below 1 ms only, which is a relatively small and acceptable value. Tested values for additive parameters δ larger than 1 ms yielded delays higher than 1 ms, so we are not reporting those tests in the Figures. The red "+" markers plotted in Figure 5 correspond to the different tested configurations.

For all four traces used in Figure 5, we can see that Dynamic Timeout achieves almost the same results as for the static coalescing. Under low load conditions η_{LPI} is slightly higher than 90% while for loads equal to $\sim$5%, $\sim$10% and $\sim$ 40% η_{LPI} is $\sim$ 50%, $\sim$ 10% and $\sim$ 1.5%, respectively. Within each Figure, it is possible to observe that there are no huge differences in the power saving and delay performance achieved under different configurations for the same traffic trace analysis (less than 10% for η_{LPI} and up to a few hundreds of μs for the delay).

We conclude that dynamically adjusting the coalescing timeout does not outperform static coalescing when the coalescing delay has to be kept low under all traffic conditions.

6.2.3 *Dynamic Queue Size*

The Dynamic Queue Size algorithm adapts N_c to the traffic conditions and slightly outperforms the other tested algorithms in terms of power saving. This behavior is shown in Figure 5, which also includes results for the Dynamic Queue Size algorithm, tested for $T_c \in [0.1, 100]\, ms$, additive parameters δ from 1 to 10 packets, and multiplicative parameters γ from 1 to 100%. Similarly to what we did for the other algorithms, we simulate EEE with coalescing using the Dynamic Queue Size algorithm over the same traffic traces as before. Each green "×" marker in Figures 5a to 5d corresponds to a different configuration for the Dynamic Queue Size algorithm.

In terms of achievable power saving, results for the Dynamic Queue Size case are, in general, slightly better than for static coalescing and for the Dynamic Timeout algorithm. Specifically, in Figure 5a, under very low load, η_{LPI} increases due to Dynamic Queue Size are relatively low. Instead, under medium/high loads (see Figures 5b and 5c) we observe an increase of a few percents in η_{LPI}, with respect to the other algorithms. Eventually, under very high loads (e.g., see Figure 5d), η_{LPI} is very small under any algorithm and configuration. In all cases, the (slightly) higher power saving is achieved at the expenses of slightly higher delay.

We can safely infer that dynamically adjusting the coalescing buffer size does not allow to achieve significantly better performance with respect to static coalescing. Indeed, having shown that static coalescing and dynamic coalescing algorithms achieve similar power saving and delay tradeoffs under a variety of traffic configurations, we conclude that static coalescers are preferable for real implementation due to their low complexity.

7. CONCLUSIONS

In this paper, we have presented a model for bidirectional EEE links with static or no coalescing. We have shown that the model can be used to accurately estimate power saving and packet delay over EEE gigabit links. This model is unique in the existing literature, in which only unidirectional EEE links are accounted for.

We have modified the ns-3 simulator to implement (*i*) the EEE standard and (*ii*) static as well as dynamic coalescing algorithms, and thus validate our model. Furthermore, we have proposed an exhaustive performance evaluation on the impact of packet coalescing techniques over EEE power saving and delay performances. Specifically, we have tested EEE and coalescing algorithms by means of real packet traces we collected at the firewall interfaces of a large web hosting center in Madrid, Spain.

Notably, we have shown that static coalescing algorithms, in which the coalescing queue size N_c and the coalescing timeout T_c are fixed, can achieve results as good as dynamic coalescing algorithms, in which either N_c and T_c can be dynamically adapted to the traffic characteristics. Hence, our model can be used to estimate the potential power saving-delay tradeoff of EEE with static or dynamic coalescing.

Our study has shown that the sole EEE standard (without coalescing) works fine under scarce traffic ($< 1\%$). In contrast, as soon as the traffic exceeds a few percents of the link capacity, EEE needs to be endowed with packet coalescing to achieve significant power saving. Thanks to coalescing, significant economy can be achieved with link loads as high as $40-50\%$, while plain EEE would not allow to achieve detectable power saving with loads higher than a few percents of the link capacity. According to our results, static packet coalescing techniques appear to be nearly-optimal, and thus preferable to dynamic techniques due to their complexity.

8. ACKNOWLEDGMENTS

This research was supported in part by the Spanish MICINN grant TEC2011-29688-C02-01.

The authors would like to thank InterHost S.A. for allowing them to collect anonymized traffic traces in their web hosting center in Madrid, Spain.

9. REFERENCES

[1] European Commission - (DG INSFO), "Study of the impacts of ICT on energy efficiency," Sept. 2008.

[2] K. Christensen, P. Reviriego, B. Nordman, M. Bennett, M. Mostowfi, and J. A. Maestro, "IEEE 802.3az: The Road to Energy Efficient Ethernet," *IEEE Communications Magazine*, vol. 48, no. 11, pp. 50–56, Nov. 2010.

[3] R. Sohan, A. Rice, A. Moore, and K. Mansley, "Characterizing 10 Gbps Network Interface Energy Consumption," in *Proceedings of IEEE LCN 2010*, Oct. 2010.

[4] IEEE Std. 802.3az, "Energy Efficient Ethernet," 2010.

[5] M. Ajmone Marsan, A. Fernandez Anta, V. Mancuso, B. Rengarajan, P. Reviriego Vasallo, and G. Rizzo, "A Simple Analytical Model for Energy Efficient Ethernet," *IEEE Communications Letters*, no. 99, pp. 1–3, June 2011.

[6] D. Larrabeiti, P. Reviriego, J. A. Hernandez, J. A. Maestro, and M. Uruena, "Towards an Energy Efficient 10 Gb/s optical Ethernet: Performance analysis and viability," *Optical Switching and Networking*, vol. 8, no. 3, pp. 131–138, Mar. 2011.

[7] S. Herrería-Alonso, M. Rodríguez-Pérez, M. Fernández-Veiga, and C. López-García, "How efficient is energy-efficient ethernet?" in *Proceedings of ICUMT*, Oct. 2011.

[8] NS-3 website: http://www.nsnam.org/.

[9] S. Herrería-Alonso, M. Rodríguez-Pérez, M. Fernández-Veiga, and C. López-García, "A GI/G/1 Model for 10Gb/s Energy Efficient Ethernet Links," *IEEE Transactions on Communications*, vol. 60, no. 11, pp. 3386–3395, Nov. 2012.

[10] M. Mostowfi and K. Christensen, "An Energy-Delay Model for a packet coalescer," in *Proceedings of IEEE Southeastcon*, Mar. 2012.

[11] P. Reviriego, J. A. Hernandez, D. Larrabeiti, and J. A. Maestro, "Performance Evaluation of Energy Efficient Ethernet," *IEEE Communications Letters*, vol. 13, no. 9, pp. 697–699, Sept. 2009.

[12] P. Reviriego, K. Christensen, J. Rabanillo, and J. A. Maestro, "Initial Evaluation of Energy Efficient Ethernet," *IEEE Communications Letters*, vol. 15, no. 5, pp. 578–580, May 2011.

[13] V. Mancuso and A. Chatzipapas, "On IEEE 802.3az Energy Efficiency in Web Hosting Centers," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1880–1883, Nov. 2012.

[14] P. Reviriego, J. A. Maestro, J. A. Hernandez, and D. Larrabeiti, "Burst Transmission for Energy Efficient Ethernet," *IEEE Computer Society*, vol. 14, no. 4, pp. 50–57, July 2010.

[15] P. Reviriego, K. Christensen, and A. Sanchez-Macian, "Using Coordinated Transmission with Energy Efficient Ethernet," in *Proceedings of IFIP Networking*, May 2011.

[16] M. Mostowfi and K. Christensen, "Saving Energy in LAN Switches: New Methods of Packet Coalescing for Energy Efficient Ethernet," in *Proceedings of IGCC*, July 2011.

[17] S. Herrería-Alonso, M. Rodríguez-Pérez, M. Fernández-Veiga, and C. López-García, "Bounded energy consumption with dynamic packet coalescing," in *Proceedings of IEEE NOC*, June 2012, pp. 1–5.

[18] B. Nordman, "EEE Savings Estimates," IEEE 802.3 Energy Efficient Ethernet Study Group, May 2007. [Online]. Available: http://www.ieee802.org/3/eee_study/public/may07/nordman_2_0507.pdf

[19] L. Kleinrock, *Queueing Systems: Theory*. John Wiley and Sons, 1975, vol. 1.

[20] T. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding data center traffic characteristics," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 92–99, Jan. 2010.

[21] Google, "(Google Data Center Video Tour)," http://www.google.com/about/datacenters/events/2009-summit.html#tab0=4, Apr. 2009.

[22] S. Bapat, "The Future of Data Centers (... and the Stuff That Goes In Them)," in *Proceedings of 1st Berkeley Symposium on Energy Efficient Electronic Systems*, June 2009.