Improving the Energy Benefit for 802.3az using Dynamic Coalescing Techniques

Angelos Chatzipapas[1,2] and Vincenzo Mancuso[1]
[1]IMDEA Networks Institute, Madrid, Spain [2]University Carlos III of Madrid, Madrid, Spain
Email: {angelos.chatzipapas, vincenzo.mancuso}@imdea.org

***Abstract*—In this work, we propose a dynamic coalescing algorithm for IEEE 802.3az standard, which dynamically adapts coalescing operations based on the current load and on the delay experienced in the link. Our results show that our algorithm almost doubles the energy efficiency of EEE with static coalescing while keeping packet delay bounded.**

***Index Terms*—802.3az; Energy Efficiency; Coalescing.**

I. Introduction

Network links in data center servers consume more than 20% of the total server energy which establishes network as the second biggest energy consumer in data centers [1]. In particular, low speed network cards, operating at 100 $Mbps$ or less, consume up to $\sim$200 mW, while 1 $Gbps$ and 10 $Gbps$ cards typically consume 5 W and 15 W, respectively, which makes reasonable the effort to reduce the energy consumption of high speed networks. Moreover, network links are heavily underutilized, i.e., 80% of the links have load less than 10% of their bandwidth [2]. Thereby, there is room for energy saving mechanism to significantly improve efficiency.

To this aim, IEEE 802.3az [3], also known as Energy Efficient Ethernet (EEE), introduces an energy saving state, namely Low Power Idle state (LPI). Recent studies [4], [5] have shown that EEE can give some energy saving benefit only under low traffic ($\leq$5%), providing that the traffic has some burstiness and the packets are large ($\sim$1500 $bytes$). Moreover, in cases of higher traffic loads, EEE's effect is almost negligible due to LPI transitioning delays. To tackle those problems, researchers have proposed packet coalescing [6] to extend the sojourn in state LPI and improve the performance of EEE at the cost of additional packet delay.

EEE with coalescing keeps longer the link in a low power state introducing a buffer of length N_c packets and a timer of duration T_c for coalescing in either direction. In contrast to the stateless legacy Ethernet, EEE with coalescing has four link states, i.e., (1) state A which is the actual packet transmission state and behaves like legacy Ethernet, (2) state LPI which is the low power state and consumes about 10% the energy of state A where the link remains there either until the link receives N_c packets in any of the link directions or until the timer T_c expires, (3) state S which is the transition $A \rightarrow LPI$ and (4) state W which is the transition $LPI \rightarrow A$.

In this paper we focus on the performance of 1 $Gbps$ EEE links with coalescing and we propose a dynamic algorithm which doubles the energy saving that can be achieved while keeping the coalescing delay within specific bounds. Although our study can be extended to EEE links operating at different speeds, we focus on 1 $Gbps$ links since they are the most commonly adopted links in nowadays data centers.

Related work. EEE with dynamic coalescing has been little investigated so far. In [7] the authors propose a dynamic coalescing queue algorithm that adapts the buffer size N_c according to the difference between an ideal energy proportional saving model and the one proposed in their paper, but it lacks of complete performance evaluation using different parameters for the dynamic queue part. Moreover the results they provide show that legacy coalescing outperforms or at least achieves results similar to the dynamic scheme. In [5] we proposed two dynamic coalescing algorithms that adapt the coalescing timer T_c and the coalescing buffer size N_c, respectively. The event that triggers the adaptation of the corresponding parameter is either the timer expiration, or the fill-up of the buffer, with no further considerations on the network performance. That paper concludes that dynamic schemes do not outperform legacy coalescing. In contrast, here we show that dynamic coalescing has potentials if based on run-time delay measurements.

II. Coalescing with Dynamic Timer Algorithm

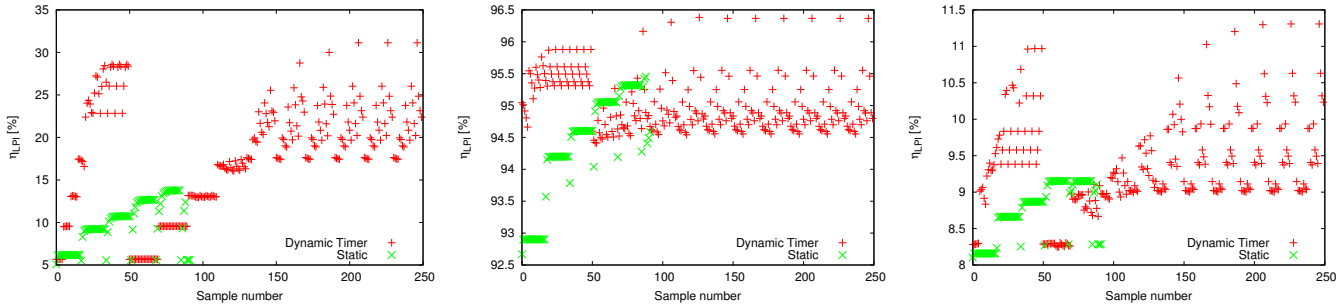
We present an algorithm whose goal is to save energy while keeping the average packet delay below a target value D_{target}.

A. Analytical approach

To enter state LPI, 1 $Gbps$ EEE links account for traffic activity in both link directions simultaneously. For such links, the model reported in [5] estimates both the energy saving, η_{LPI}, and the delay in each link direction, D_i, with and without coalescing. Using the model in [5] we have performed a sensitivity analysis of energy saving and coalescing delay with respect to N_c and T_c. Based on that, we have observed that it is enough to adjust only the coalescing timer to achieve almost optimal results while providing delay guarantees. Therefore, our proposal uses the sensed delay as control signal to trigger the dynamic adaptation of the coalescing timer.

B. Algorithm

Our dynamic coalescing timer algorithm estimates the average delay D_i, using a low-pass filter. When the transition from state C to state W happens, the algorithm adapts the T_c value (reduce or increase) based on the experienced average delay and D_{target}, as reported in Algorithm 1.



(a) $\rho_1 = 0.66\%$, $\rho_2 = 10.5\%$. (b) $\rho_1 = 0.2\%$, $\rho_2 = 0.06\%$. (c) $\rho_1 = 1.0\%$, $\rho_2 = 40\%$.

Fig. 1: η_{LPI} achieved using dynamic and static coalescing algorithms. The results shown in the subfigures have been obtained by applying several configurations of the coalescing parameters for various traffic conditions and subject to $D_{target} \leq 1\ ms$.

Algorithm 1 Dynamic Coalescing Timer Algorithm

procedure DYNTIMER($\alpha, T_c^{min}, T_c^{max}, N_c, D_{target}$)
  while Transition C $\rightarrow$ W **do**
    if $(D1 \&\& D2) \leq D_{target}$ **then**
      if $T_c < T_c^{\max}$ **then**
        increase T_c by α
      else
        $T_c = T_c^{\max}$
      end if
    else
      if $T_c > T_c^{\min}$ **then**
        decrease T_c by α
      else
        $T_c = T_c^{\min}$
      end if
    end if
    restart the coalescing timer with the updated T_c
  end while
end procedure

T_c is increased in the algorithm when the sensed delay is lower than D_{target}, and vice versa it is reduced when the delay is excessive. The result is twofold: (i) given that T_c is adapted during state transition $C \rightarrow A$, the process of delay adaptation occurs at the timescale of milliseconds, which permits to promptly react to changes in packet delay; (ii) our dynamic algorithm can adapt to any change in traffic conditions from low to high load and vice versa just by estimating the packet delay. Since load variations happen very often during the day, our simple adaptation mechanism can produce great benefits.

In our dynamic coalescing algorithms, additive or multiplicative increases and decreases can be used, or a combination of both. In the decryption reported in Algorithm 1, we simply use parameter α to indicate that an increase or decrease happens, and in our experiments we have tested additive and multiplicative options in all combinations.

It is important to note that in 1 $Gbps$ EEE links the two directions are correlated and therefore the algorithm has to measure the packet delay in the two directions at the same time, and both delays have to be kept below D_{target}.

C. Evaluation

In Fig. 1, we show the achievable energy savings obtained by different configurations of static schemes (with fixed N_c and T_c values), and delay-bounded dynamic schemes (with fixed N_c and α values) for three representative load combinations (ρ_1, ρ_2). We have collected real traffic traces in a large web hosting center in Madrid and we imported those traces in a modified version of NS-3[1] simulator to simulate the behavior of EEE and coalescing algorithms for 1 $Gbps$ links with realistic traffic traces.

The best results achieved with the static algorithm in any of the depicted scenarios are very far from the best results of the dynamic scheme. Indeed, delay-bounded dynamic schemes practically double the gain achieved by static algorithms.

III. Conclusion

In this paper we have presented a dynamic coalescing algorithm for EEE . Our algorithm adapts its coalescing timer according to the delay sensed by packets in the link. The resulting algorithm can double the energy saving benefit with respect to legacy coalescing schemes.

References

[1] J. Arjona Aroca, A. Chatzipapas, A. Fernández Anta, and V. Mancuso, "A measurement-based analysis of the energy consumption of data center servers," in Proceedings of the 5th International Conference on Future Energy Systems (ACM e-Energy '14), Jun. 2014, pp. 63–74.

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

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

[4] 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.

[5] A. Chatzipapas and V. Mancuso, "Modelling and real-trace-based evaluation of static and dynamic coalescing for energy efficient ethernet," in Proceedings of the Fourth International Conference on Future Energy Systems (ACM e-Energy '13), May 2013, pp. 161–172.

[6] 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.

[7] S. Herrería-Alonso, M. Rodríguez-Pérez, M. Fernandez-Veiga, and C. López-García, "Bounded energy consumption with dynamic packet coalescing," in 17th European Conference on Networks and Optical Communications (NOC), 2012, pp. 1–5.

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