Bounded Energy Consumption with Dynamic Packet Coalescing

Sergio Herrería-Alonso, Miguel Rodríguez-Pérez, Manuel Fernández-Veiga, and Cándido López-García

Telematics Engineering Department
University of Vigo
36310 Vigo, Spain
{sha,miguel,mveiga,candido}@det.uvigo.es

Abstract—Switching the operating state on machines is probably the simpler way to reduce the energy consumption of otherwise non energy-aware devices. Indeed, many energy-aware systems allow two states of operation: a low power idle (LPI) mode, in which the system is unable to process data, and a fully working mode. For networking equipment, packet coalescing is the best known algorithm for managing the LPI mode and has been proved able to get power savings nearing theoretical limits at the expense of packet delay when properly tuned. However, in real networks with non stationary traffic conditions, static tuning induces higher than necessary delays. In this paper we present the first dynamic algorithm that adapts packet coalescing configuration parameters to the real time traffic characteristics. Simulation results show that our algorithm is capable of achieving a given target energy efficiency while keeping packet delay as low as possible.

Index Terms—Energy efficiency, low power idle, energy-delay tradeoff, packet coalescing.

I. INTRODUCTION

Power requirements of networking infrastructure have been rising steadily during the last few years. The growing demands are caused by a variety of motives: augmented nominal link bandwidths, new link deployments, growing number of devices connected to the Internet, etc. These energy needs are worrisome for two reasons. Firstly, they are a new cause of environmental concern and, at secondly, they increment the operational expenses of network operators. The existence of energy-constrained devices and the aforementioned concerns are spurring the search for more energy-friendly designs for networking equipment; the final goal being devices whose power needs escalate linearly with traffic load [1], [2].

Networking equipment usually provides several energy profiles with different performance capabilities. The operational status of these devices are governed with the help of energy-aware algorithms that can take into account the current traffic conditions. The simpler controlling algorithms usually employ just both ends of the spectrum of available energy profiles, that is the most energy demanding and performant one, and a power save mode incapable of transmitting traffic. The latter is usually known as low power idle mode, or LPI for short. Probably, the most natural way to govern the LPI mode is entering it whenever the transmission queue becomes empty and restoring normal operation when new traffic arrives. Although, at a first look, it may seem that such a straightforward approach to energy management would produce mediocre results, it generates great power savings when the traffic is bursty in nature and the switching times between the active and LPI states are relatively short.

Unfortunately, these controllers do not yield good results when the traffic is not bursty enough. Under these traffic conditions, even if the transition times between operating modes are fast, the device spends most of the energy on switching between states rather than on useful data transmission. This is why more advanced controllers employ a technique called packet coalescing or burst transmission, by which a batch of packets is assembled while the device rests in LPI mode. This approach has been used with great success in several transmission technologies, such as Ethernet [3], [4], [5], WiFi [6], [7] and EPON [8], since it significantly reduces the number of state transitions thus improving energy efficiency, albeit at the expense of increased packet delay.

Packet coalescers have two configuration parameters to decide when to exit the LPI mode: a queue threshold and a timeout since the first packet arrives to the transmission queue, with both parameters having effects on packet delay and energy savings. Obviously, a good tuning of the queue threshold is key for the performance of the mechanism [9]. If the threshold is too high, packets can be excessively delayed. On contrary, setting a low threshold results in less power savings. An additional problem is that a single threshold value does not suit well for any possible incoming traffic. For instance, when traffic load is low, increasing the threshold, even from modest values, produces marginal increments in power savings. Under these circumstances, a low threshold is desirable, as it provides small latencies and good enough energy savings. The fundamental reason is that packet interarrival times are so long that the time needed to reach the threshold, and thus in LPI mode, gets much longer than the transition times, making them almost irrelevant in terms of power usage. For high traffic loads, the situation is just reversed. If the threshold were not increased, the time waiting to reach it would be in the order of the transition times, greatly diminishing the obtained power savings.

Throughout this paper we present a new dynamic method for adjusting the queue threshold used with packet coalescing to the real time incoming traffic characteristics. The method makes no assumption whatsoever about the incoming traffic
pattern. Simulation results carried out with both synthetic and
real traffic traces show that our mechanism is able to achieve
a pre-established target for energy consumption while keeping
the packet delay as low as possible.

The rest of the paper is organized as follows. In Section II
we develop the mathematical model on which the method is
later built. Section III explains the proposed method and the
experimental results are shown in Section IV. Finally, we lay
down our conclusions in Section V.

II. POWER ANALYSIS

In this section we develop a simple analysis to quantify the
power consumption of an energy-aware networking system.
Figure 1 shows an example of how the transmission buffer
evolves when using packet coalescing. Let $T_s$ be the time
employed to coordinate the status change to LPI and $T_w$
the time needed to awake the device. Time intervals when
packets are not being transmitted are inactive periods and
may be composed of the two transition periods, $T_s$ and $T_w$,
and a sleeping period $T_{off}$ in which the device stays in LPI.
As dictated by the coalescing algorithm, the sleeping system
will awake when the amount of queued packets reaches the
queue threshold $Q_w$ although, to avoid excessive delays, the
maximum time a device can be in the LPI mode continuously
is limited to $T_{max}$ since the first packet is buffered. The time
interval when the device is transmitting packets is the busy
period $T_{on}$. Then, an inactive period followed by a busy period
forms a busy cycle $T_{cycle}$.

Let $P$ be the mean power consumed when the device is
active and $P_{cycle}$ the mean power consumption on a given
busy cycle. Obviously, $P_{cycle}$ depends on the amount of time
the device spends in each possible state:

$$P_{cycle} = \frac{T_{off}P_{off} + (T_s + T_w)P + T_{on}P}{T_{cycle}},$$

where $P_{off}$ is the mean power consumed in the LPI mode.
Note that it is assumed that the device consumes about the
same power during transitions as in the active state since
it is expected that many components of the device have to
be operative during the state changes. Immediately, the ratio
between the energy consumed on an energy-aware device and
that consumed on a device that is always operating in the
active mode is given by

$$\varphi = \frac{P_{cycle}}{P} = \frac{\varphi_{off}T_{off} + T_s + T_w + T_{on}}{T_{cycle}},$$

where $\varphi_{off} = P_{off}/P$ is the portion of the active mode energy
consumption demanded when the device is in LPI.

We will compare this consumption against that on an
idealized device with optimum performance. Such a system
would waste no energy in transitions, staying idle for all
the time there is no traffic to be processed. Therefore, this
idealized device would just consume:

$$\varphi_{ideal} = \frac{\varphi_{off}(T_{cycle} - T_{on}) + T_{on}}{T_{cycle}}.$$

Then, from (2) and (3), and substituting $T_{cycle} = T_{off} + T_s +
T_w + T_{on}$, the difference between the ideal power savings and
those obtained on a particular energy-aware device is:

$$\eta = \varphi - \varphi_{ideal} = (1 - \varphi_{off})\frac{T_s + T_w}{T_{off} + T_s + T_w + T_{on}}.$$

III. DYNAMIC PACKET COALESCING

This section presents a method to dynamically accommodate
the $Q_w$ parameter to incoming traffic. Fig. 2 illustrates
the trade-off between the average packet delay and the power
consumption for different offered traffic loads $\rho$. Clearly,
energy efficiency can only be improved at the expense of
increasing packet delay. However, it should be noted that,
when power demands are near the minimum ideal consump-
tion, even small additional consumption reductions involve
unacceptable large increments on packet delay. Our goal here
is to keep $\eta = \varphi - \varphi_{ideal}$ bounded while minimizing packet
delay, adapted to current traffic conditions.

As it can be seen in (4), to adjust the power consumption
on an energy-aware system, we can only control $T_{off}$ and $T_{on}$,
as $T_s$, $T_w$ and $\varphi_{off}$ are usually a given for any specific device.
Modifying the $Q_w$ parameter is the simplest way to influence
on $T_{off}$. Note that the total length of $T_{off}$ can be split up

![Fig. 1. Busy cycle when using packet coalescing.](image1)

![Fig. 2. Relation between average packet delay and power efficiency for different offered traffic loads.](image2)
Dynamic packet coalescing algorithm:

1) At the end of cycle $i$ measure $T_{on}[i]$ and $T_{off}[i]$.
2) Compute $\eta[i]$ according to (4).
3) Update $Q_w$ as shown:

$$Q_w[i + 1] \leftarrow \begin{cases} Q_w[i] + \gamma & \text{if } \eta[i] > \eta^* \max \{Q_w[i] - \gamma, 0\} & \text{if } \eta[i] < \eta^* \end{cases}$$

in any other case.

Table I

<table>
<thead>
<tr>
<th>Qw[1] + 1</th>
<th>Qw[i] + γ if η[i] &gt; η*</th>
<th>max[Qw[i] - γ, 0] if η[i] &lt; η*</th>
</tr>
</thead>
</table>

into two parts. The first accounts for the time until the first packet arrival while idle and can be even zero if it arrives while the device is still transitioning into LPI. The second one is the waiting time for the other $Q_w - 1$ packets to appear before waking up the device back. Clearly, there is no way to influence on the first part, but we can increase (or decrease) the second one simply incrementing (or decrementing) $Q_w$.

Additionally, $Q_w$ has also a direct effect on $T_{on}$. The busy period is directly influenced by the incoming traffic rate and by the length of the transmission buffer when exiting the LPI mode. Therefore, modifying $Q_w$, that is, the buffer length at wake up time, also entails varying $T_{on}$ in the same direction. Actually, when the device restores its normal operation, the length of the transmission buffer will be $Q_w$ plus the number of packets that arrived while transitioning to active.

The main purpose of our algorithm is to find the minimum $Q_w$ value that satisfies a maximum $\eta$ value, $\eta^*$. This way we can achieve our two goals: satisfy the power efficiency target and minimize packet delay. We assume that, although traffic characteristics do change over time, they usually do so over periods of several busy cycles. So, to adjust the $Q_w$ parameter to the existent traffic conditions, we measure the performance in the current busy cycle. That is, if the $i$th cycle is the current one, we indirectly calculate $\eta[i]$ by measuring $T_{on}[i]$ and $T_{off}[i]$, and compare it with our target $\eta^*$. If $\eta[i] > \eta^*$ the device is consuming more power than desired, so $Q_w$ should be incremented to increase both $T_{on}$ and $T_{off}$. Conversely, if $\eta[i] < \eta^*$ the current power consumption is low enough and $Q_w$ can be reduced to diminish packet delay. Assuming that traffic characteristics in cycle $i + 1$ remain similar to those of cycle $i$, we propose to modify $Q_w$ by just a small amount $\gamma$ according to the $\eta[i]$ value as shown in Table I. The stability of this dynamic algorithm is proved in the Appendix.

Finally, our algorithm preserves the $T_{max}$ timer since it is just a safety measure that prevents packet delay to get uncontrolled in the case of a sudden change of traffic characteristics.

IV. Evaluation

To validate our algorithm we have chosen to apply it on IEEE 802.3az Ethernet interfaces [10] since packet coalescing can be successfully employed to manage the LPI mode defined for this kind of interfaces [3], [4]. We conducted several simulation experiments on the ns-2 simulator [11]. The simulated interface has a 10 Gb/s capacity and packet arrivals follow a Pareto distribution with an average arrival rate $\lambda$ varying from 10 Mb/s to 10 Gb/s. This is one of the most common heavy-tailed distributions and it can be successfully used for characterizing self-similar Internet traffic. The Pareto distribution is configured with a shape parameter $\alpha = 2.5$ and the packet size is set to 1500 bytes.

We chose the popular 10GBase-T interface for our experiments, so $T_s = 2.88 \mu s$, $T_w = 4.48 \mu s$ and $\varphi_{off} = 0.1$ according to several estimates provided by different manufacturers during the standardization process of the IEEE 802.3az standard.

We first configured standard packet coalescing with $T_{max} = 100 \mu s$ and $Q_w = 25$ packets. This $Q_w$ value has been experimentally selected to guarantee a reasonable maximum energy consumption of $\eta^* = 5\%$ over the ideal minimum consumption in all the simulated scenarios. Naturally, our method was also configured with the same $\eta^*$ and $T_{max}$ values to ensure a fair comparison. Finally, we set $\gamma = 1$ packet as a good compromise between $Q_w$ stability and a fast response to changing traffic characteristics.

We ran each simulation for 10 seconds and measured both energy consumption and average queueing delay. Each simulation experiment was repeated ten times using a different seed value when initializing the random number generator. Then, an average of the measured parameter was taken over all runs.

Figure 3(a) shows the energy consumed in the simulated interface without packet coalescing, with standard coalescing and with dynamic coalescing. Additionally, the minimum theoretical consumption and the target consumption (that is, the theoretical minimum plus 5%) are also depicted in the graph. As expected, packet coalescing provides greater energy savings at medium and high traffic loads since it significantly reduces the number of transitions between the LPI and active modes. Also note that dynamic coalescing increases energy consumption in a hardly perceivable way maintaining power demands around (or below) the target consumption for any load. Figure 3(b) shows the average queueing delay obtained with the three analyzed methods. As expected, the queueing delay is very short without packet coalescing. However, with standard coalescing under low load, the delay is maximum and equal to $T_{max} + T_w = 104.48 \mu s$. Note that, with Pareto distributions, the minimum interarrival time is set to $(\alpha - 1)/(\alpha \lambda)^{-1}$. Then, for $\lambda < 72$ Mb/s, this value is greater than $T_{max}$ and, therefore, the sleeping interface will always resume its normal operation after the $T_{max}$ timeout expires. On contrary, our proposal successfully adjusts the $Q_w$ parameter to the different load conditions providing much shorter

---

1 In [5] it is proved that, for a M/G/1 queue with $B$ backlogged packets, $T_{on}$ is equal to that of a normal M/G/1 queue plus half the service time of $B - 1$ packets.

2 Note that we can only influence on the length of individual busy periods. The overall $T_{on}$ time remains fixed since it only depends on traffic load.

3 Although 95% confidence intervals have been also calculated, they are not represented in the graphs since they were consistently lower than ±1% and just cluttered the figures.

4 That is, the interface is awaken as soon as a new packet is ready for transmission ($Q_w = 1$ packet).
delays, especially at low loads, at the expense of slightly increasing power consumption. We repeated this experiment with $T_{\text{max}} = 1\text{ ms}$. As shown in Fig. 4, dynamic coalescing achieves in this scenario more significant reductions in packet delay while maintaining power demands around the configured target.

We also evaluated the performance of our algorithm with narrower ($\eta^* = 1\%$) and wider ($\eta^* = 10\%$) margins over the ideal power consumption. Figure 5(a) shows the percentage of extra energy consumed compared to the minimum ideal consumption for the different $\eta^*$ values. As expected, our proposal can effectively bound the peak consumption of an energy-aware device although, obviously, greater power savings are obtained at the penalty of increasing packet delay, as shown in Fig. 5(b).

Finally, we validated our algorithm using real world traffic traces publicly available from the CAIDA archive [12]. The analyzed CAIDA traces were collected during 2011 on a 10 Gb/s backbone Ethernet link of a Tier1 ISP between Chicago, IL and Seattle, WA. We configured coalescing algorithms with $T_{\text{max}} = 1\text{ ms}$ and $\eta^* = 5\%$. Table II shows the obtained results. As with Pareto traffic, dynamic coalescing reduces packet delay while maintaining power consumption around the configured target.

V. CONCLUSIONS

This paper presents a promising variant of the well-known packet coalescing algorithm proposed to manage the operating state of energy-aware networking equipment. The algorithm dynamically adjusts coalescing configuration parameters in accordance with current traffic conditions sparing the administrator from manually tuning them. Simulation results for both synthetic traffic and real Internet traffic traces confirm that the new method is able to keep energy consumption bounded while minimizing packet delay.

ACKNOWLEDGMENTS

This work was supported by the “Ministerio de Educación y Ciencia” through the project TEC2009-12135 of the “Plan Nacional de I+D+I” (partly financed with FEDER funds).

<table>
<thead>
<tr>
<th>Mean Rate (Gb/s)</th>
<th>Energy (%)</th>
<th>Delay (\mu s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.30</td>
<td>67.2</td>
<td>4.12</td>
</tr>
<tr>
<td>0.44</td>
<td>60.5</td>
<td>4.03</td>
</tr>
<tr>
<td>0.56</td>
<td>63.8</td>
<td>4.01</td>
</tr>
<tr>
<td>1.18</td>
<td>69.4</td>
<td>3.99</td>
</tr>
<tr>
<td>1.39</td>
<td>73.2</td>
<td>3.97</td>
</tr>
<tr>
<td>1.51</td>
<td>74.9</td>
<td>3.95</td>
</tr>
<tr>
<td>1.91</td>
<td>80.2</td>
<td>3.92</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mean Rate (Gb/s)</th>
<th>Energy (%)</th>
<th>Delay (\mu s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.30</td>
<td>14.9</td>
<td>17.0</td>
</tr>
<tr>
<td>0.44</td>
<td>19.9</td>
<td>18.7</td>
</tr>
<tr>
<td>0.56</td>
<td>24.2</td>
<td>18.8</td>
</tr>
<tr>
<td>1.18</td>
<td>27.3</td>
<td>25.4</td>
</tr>
<tr>
<td>1.39</td>
<td>26.3</td>
<td>24.2</td>
</tr>
<tr>
<td>1.51</td>
<td>27.6</td>
<td>26.3</td>
</tr>
<tr>
<td>1.91</td>
<td>31.6</td>
<td>30.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mean Rate (Gb/s)</th>
<th>Energy (%)</th>
<th>Delay (\mu s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.30</td>
<td>139.3</td>
<td>17.4</td>
</tr>
<tr>
<td>0.44</td>
<td>98.0</td>
<td>18.7</td>
</tr>
<tr>
<td>0.56</td>
<td>88.8</td>
<td>19.9</td>
</tr>
<tr>
<td>1.18</td>
<td>69.4</td>
<td>27.3</td>
</tr>
<tr>
<td>1.39</td>
<td>66.2</td>
<td>28.4</td>
</tr>
<tr>
<td>1.51</td>
<td>57.1</td>
<td>28.4</td>
</tr>
<tr>
<td>1.91</td>
<td>52.4</td>
<td>28.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mean Rate (Gb/s)</th>
<th>Energy (%)</th>
<th>Delay (\mu s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.30</td>
<td>114.4</td>
<td>17.7</td>
</tr>
<tr>
<td>0.44</td>
<td>61.6</td>
<td>20.0</td>
</tr>
<tr>
<td>0.56</td>
<td>60.9</td>
<td>25.6</td>
</tr>
<tr>
<td>1.18</td>
<td>56.6</td>
<td>25.6</td>
</tr>
<tr>
<td>1.39</td>
<td>55.5</td>
<td>27.4</td>
</tr>
<tr>
<td>1.51</td>
<td>54.9</td>
<td>28.6</td>
</tr>
<tr>
<td>1.91</td>
<td>52.4</td>
<td>32.2</td>
</tr>
</tbody>
</table>
derivative, this system reaches the fixed equilibrium point $Q_w^*$ when
\begin{equation}
    f(Q_w^*) = (T_s + T_w) \left( \frac{1 - \varphi_{\text{off}}}{\eta^*} - 1 \right)
\end{equation}
which has a unique solution if $f(1) \leq (T_s + T_w)((1 - \varphi_{\text{off}})/\eta^* - 1)$, that is, $Q_w^* \geq 1$. Otherwise, the target energy consumption $\eta^*$ is not achievable. Then, assuming that this condition holds, (5) can be expressed as
\begin{equation}
    Q_w[i + 1] = Q_w[i] + \gamma \sgn \left( \frac{(1 - \varphi_{\text{off}})(T_s + T_w)}{T_s + T_w + f(Q_w[i])} - \eta^* \right) 
\end{equation}
\begin{equation}
    \simeq Q_w[i] + \gamma \frac{2}{\pi} \arctan \left[ u \left( \frac{(1 - \varphi_{\text{off}})(T_s + T_w)}{T_s + T_w + f(Q_w[i])} - \eta^* \right) \right].
\end{equation}
The approximation becomes exact in the limit as $u \to \infty$.

It is straightforward to prove via linearization that this system is stable if the derivative of (7) at the equilibrium point has an absolute value strictly less than one, that is,
\begin{equation}
    \left| 1 - \gamma \frac{2u}{\pi(1 - \varphi_{\text{off}})(T_s + T_w)} f'(Q_w^*) \right| < 1.
\end{equation}
Since $f'(\cdot)$ is positive and bounded, there always exists some $\gamma$ sufficiently small that makes that this condition is met.

**References**


