INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. (2015)
Published online in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/dac.2934

Energy efficiency of TCP: An analytical model and its application to reduce energy consumption of the most diffused transport protocol

Muhammad Usman[1,*,†], Dzmitry Kliazovich[2], Fabrizio Granelli[1], Pascal Bouvry[2] and Piero Castoldi[3]

[1]*University of Trento, Italy*
[2]*University of Luxembourg, Luxembourg*
[3]*Scuola Superiore Sant'Anna, Italy*

SUMMARY

Energy efficient communications become a challenge for both industries and researchers. Incorporating energy efficiency into the design of network protocols and architectures represents a relevant issue in networking research. Currently, very few works address energy efficiency as a fundamental feature of network protocols. This paper benchmarks energy efficiency of TCP to understand the parameters and operational mechanics that determine and contribute to energy consumption. We propose an analytical model with energy consumption to protocol operation cycles and novel optimization techniques for reducing energy consumption of TCP. The evaluation results, obtained from NS2 simulations, demonstrate that even minor modifications of the protocol behavior can bring significant savings of energy. Copyright © 2015 John Wiley & Sons, Ltd.

Received 22 October 2014; Revised 11 December 2014; Accepted 8 January 2015

KEY WORDS: green ICT; network protocols; TCP; energy efficiency

1. INTRODUCTION

Carbon footprint of the Internet has become a great concern for the ICT sector. Improving energy efficiency is one of the most cost effective ways to reduce such carbon footprint. ICT sector is estimated to consume 8% to 10% of EU's electricity production and is overall responsible for 2.5% to 4% of Green House Gases (GHG) emission [1]. GHG not only affects the environment, but has a serious impact on the economy too. ICT currently corresponds to 2% of a worldwide carbon emission and this number is increasing at a compound rate of 6% annually [2]. One Terawatt hours consumption of electricity in ICT sector produces approximately 0.75 million tons of CO_2 [3]. European Union published a report stating that 15% to 30% decrease in carbon footprint is required to avoid a 2°C increase in global temperature [4]. Telecommunication networks are responsible for 30% to 37% of GHG emission produced by ICT [5, 6]. In addition, reducing energy consumption leads to economic competitiveness in the fact of constantly increasing energy prices. The operation of large geographically distributed data centers requires significant amount of energy contributing a large part of operational cost of cloud data center [7, 8]. Energy consumption takes 25% of the operational cost (OPEX) of data center, and it is expected to increase to 50% in next few years [9]. Impact of ICT where possible and to create greener environment by 'greening' ICT systems [10].

*Correspondence to: Muhammad Usman, University of Trento, Italy.
†E-mail: muhammad.usman@unitn.it

In recent years, productive efforts have been devoted to cut down redundant energy consumption in ICT sector. These efforts are usually termed as 'greening the internet'. Antonopoulos *et al*. [11–13] proposed different MAC layer solutions to reduce energy consumption of network nodes. However, a very few research works addressed energy efficiency of Transmission Control Protocol (TCP). Current techniques for energy efficiency can be classified into the following four categories: (i) resource consolidation, (ii) virtualization, (iii) selective connectedness and (iv) proportional computing [14]. Bianzino *et al*. [provides taxonomy of green networking research showing that only two approaches focus on transport layer protocols like TCP. Most of the solutions work on data link and network layers focusing on network technologies rather than protocols. Wang *et al*. [15] focused on computation cost of TCP at sender node with no detailed analysis of TCP protocol energy consumption. Irish *et al*. [16] proposed a sleep mode option to keep the client server connection logically alive and enabling larger number of PCs to remain Energy Star enabled . The proposal in [16] does not work to calculate sleep time during TCP operation, but rather than that, it provides a way of operation when the server receives sleep flag form client.

TCP is still the most used transport layer protocol today carrying more than 90% of all internet traffic including almost all kind of foreground and background traffic like YouTube and Netflix [17, 18]. A recent survey shows that YouTube and Netflix contribute more than 50% of all North America's internet traffic [19]. As a consequence, making TCP energy efficient can save significant amount of energy on the internet and mobile devices, where evolution of battery technology is too slow as compared to the increase in energy consumption [20].

In this paper, we first propose an analytical model relating energy consumption of TCP flow (in congestion avoidance phase) to protocol cycles. In practice, TCP sends traffic in bursts (window) and then it waits for acknowledgements from receiver before sending the next congestion window. The analytical model keeps track of every packet and calculates energy consumption of TCP flow as a summation of energy consumed during the transmission time and the idle time. In the paper, we consider energy consumption of TCP equals to the energy a node consumes to process a TCP flow.

Based on such model, we propose a green TCP solution which focuses on relating energy with protocol operational cycle. TCP, during each protocol cycle, spends some time in sending packets and then remains idle in waiting for acknowledgements. Nodes consume energy during both phases of protocol cycle. Our contribution is to model this energy consumption of node for each protocol cycle and then to propose specific modifications to the standard TCP algorithm to improve its energy efficiency. The proposal includes: (i) adding sleeping flags in TCP header, (ii) reducing inter packet gaps due to bottleneck link between sender and receiver (iii) delaying acknowledgements.

(a) *Sleep flag in TCP header*: A sleeping flag is added to the last packet of each congestion window informing subsequent nodes that the sender has no data to send for some time—therefore, they may switch to sleep mode. The nodes can decide whether to sleep or not depending upon the number of current TCP flows and the available sleeping time itself—if this is sufficiently greater than transition time. Transition time is the time required by nodes or ports to go to sleep and wake up. Different flows can be aligned together to increase the sleep time without a relevant impact on TCP throughput. This alignment process is called synchronization of flows. Turning a specific port of a node to sleep can also save a significant amount of energy. In current commercial switches the power of a single port is 0.75 W in an 8 port switch and is almost double 1.375 W in 48 ports switch used in data center access network [16]. Similar is the case for putting Network Interface Card (NIC) to sleep mode. Average load of NIC is 30% in servers, and in computers NIC takes 5% of total load [21–23].
(b) *Reducing inter packet gaps*: Due to bottleneck links between sender and receiver acknowledgments arrive with small delays resulting in same gaps between data packets. Such inter packets gaps can be merged together by buffering data packets and generating longer sleeping time intervals.
(c) *Delayed acknowledgements*: Experiments show that delaying acknowledgment can also save significant amount of energy.

The rest of the paper is organized as follows. Section II describes a model for the energy consumption of TCP/IP protocol and illustrates different methods to save energy. Results and discussions about simulation and analytical model are provided in Section III. Finally, Section IV presents the conclusion along with future directions.

DOI: 10.1002/dac

2. "GREEN" TCP PROPOSALS

The goal of this section is (i) to understand and model energy consumption dynamics of legacy TCP protocol and (ii) to introduce techniques capable of improving energy efficiency of the transport protocol.

A. Energy consumption model of -TCP

TCP is currently the most widely used transport protocol in internet mainly due to reliable communication between end systems and flow control [24]. Reliability is achieved by requiring the receiver to positively acknowledge successfully received segments of information. These acknowledgements also help the sender to adapt its transmission rate better matching the available end-to-end network capacity. In a steady state, during congestion avoidance, TCP conservatively increases the contention window by one segment every RTT. This increase continues until the first packet loss detected by the reception of three duplicate acknowledgements. We call this period a TCP round.

Several models have been proposed to address energy consumption of TCP. However, most of them work only for a specific scenario. Hu *et al*. [25] proposed a model for the energy consumption of TCP traffic over wireless networks with cooperative relaying. Ayadi *et al*. [26] proposed a model for energy consumption of TCP in low power lossy networks where segment size can be tuned to optimize energy consumption. Hashimoto *et al*. [27] proposed a model for energy consumption of TCP with burst transmission over wireless networks. Mathis *et al*. [28] proposed a model for macroscopic behavior of TCP.

To account the length of inactivity gaps in each RTT and the energy consumption of network nodes, we first compute the number of TCP packets transmitted in one round:

$$\frac{\mathrm{W}}{2}+\left(\frac{\mathrm{W}}{2}+\frac{1}{\mathrm{r}}\right)+\left(\frac{\mathrm{W}}{2}+\frac{2}{\mathrm{r}}\right)+\ldots\ldots+\left(\frac{\mathrm{W}}{2}+\frac{\mathrm{N}}{\mathrm{r}}\right)=\sum_{\mathrm{n}=0}^{\frac{\mathrm{rW}}{2}}\left(\frac{\mathrm{W}}{2}+\frac{\mathrm{n}}{\mathrm{r}}\right), \tag{1}$$

where W is a maximum achievable size of congestion window, r is the average number of data packets acknowledged by a single acknowledgement packet at the receiver (accounts for delayed acknowledgement strategy) and N is number of RTTs in one round. In case of delayed acknowledgements, one round is equal to $rW/2$ RTTs.

Solving Equation (1) we obtain the number of packets sent during one round

$$=\frac{3\mathrm{rW}^2}{8}+\frac{3\mathrm{W}}{4}. \tag{2}$$

Now dividing Equation (2) by number of RTTs in one round, we obtain average throughput in packets per RTT:-

$$\mathrm{T}=\frac{\frac{3\mathrm{rW}^2}{8}+\frac{3\mathrm{W}}{4}}{\frac{\mathrm{rW}}{2}}=\frac{3\mathrm{W}}{4}+\frac{3}{2r}. \tag{3}$$

Equation (3) can further be re-written as a function of congestion probability $\mathrm{P_c}$, or the probability of having one packet loss in one round:-

$$\mathrm{P_c}=\frac{1}{\frac{3\mathrm{rW}^2}{8}+\frac{3\mathrm{W}}{4}}=\frac{8}{3\mathrm{rW}+6\mathrm{W}}. \tag{4}$$

In addition to packet losses due to congestion, we also consider link errors. As a result, the total error probability is given by

$$P=\frac{P_l}{\left(1-e^{\left(\frac{-P_l}{P_c}\right)}\right)}, \tag{5}$$

where P_l is a probability of having one data packet corrupted during its transmission over the network. Packet error rate can be derived from the bit error rate of network link. Assuming independent and uniformly distributed bit errors:

DOI: 10.1002/dac

$$P_{PER} = 1\text{-}(1\text{-}P_{BER})^{M},$$

where M is number of bits in a packet. *"M* is equal to $1500 \times 8 = 12\,000$ bits for the most common Ethernet MTU.

The $3W/4$ component in Equation (4) is very small if compared to $3rW^2/8$, and it can be neglected. Then, the maximum achievable TCP window can be expressed in terms of *P*:

$$W = \sqrt{\frac{8}{3rP}} \tag{6}$$

and average throughput in Equation (3) can be written in terms of *P* by putting Equation (6) in Equation (3).

$$T = \frac{3MSS}{4RTT}\sqrt{\frac{8}{3rP}}\left[\frac{bytes}{s}\right] = \frac{12MSS}{RTT}\sqrt{\frac{2}{3rP}}\left[\frac{bits}{s}\right]. \tag{7}$$

Equation (7) provides a deterministic model of average TCP throughput where P is average packet error probability due to both congestion and link errors, and MSS is the maximum segment size of TCP in bytes. Assuming that *P* for a given link is known W can be found using Equation (6).

Dynamic Shutdown (DNS) is one possible power saving mode in network devices that turns network node to sleep during intervals when it is not processing any data. The approach to develop energy consumption model is to find idle intervals in the mechanics of steady state TCP. TCP is designed to provide in order and lossless segment delivery for reliable communication. TCP does so by creating burst traffic as illustrated in Figure 1. The burst nature of TCP creates inactivity intervals during algorithm progression. Dynamic Shutdown (DNS) can be applied to network switches/routers or their components during these gaps of inactivity in order to save energy.

Figure 1 shows the general behavior of TCP in steady state during each round trip time. For reliable communication between two hosts, TCP follows a sliding-window based congestion control scheme, where new packets can be transmitted only when old ones are acknowledged. In steady state, the congestion window starts with half of the maximum achievable window size W. TCP increases congestion window (cwnd) by one and transmits all segments in burst of cwnd upon reception of acknowledgments for all segments sent in last RTT, and it waits for the corresponding acknowledgments. This generates bursty traffic in the network, but also idle periods while TCP is waiting for acknowledgements.

Let T_{Go}, $T_{G_1}T_{G_2}T_{G_2}$ and T_{G_n} be the time intervals when TCP is waiting for acknowledgements (and not sending TCP packets).

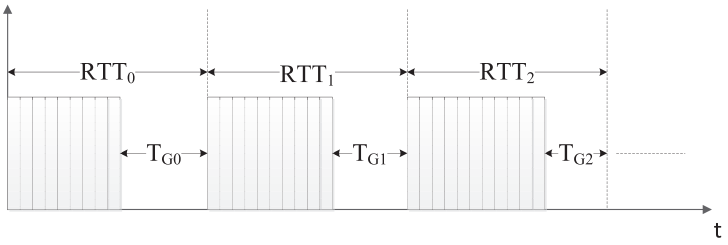

Figure 1. Gaps between bursts of packets.

DOI: 10.1002/dac

Such gap intervals are modeled by the following formula:

$$\begin{aligned}\text{TIME spent in Gap} &= \text{RTT-Packets Transmission Time}\\ \mathrm{T_{G0}} &= \mathrm{RTT_0}\text{-}\left(\frac{\mathrm{W}}{2}\right)\times\frac{\mathrm{M}}{\mathrm{D}} \quad [\mathrm{s}],\\ \mathrm{T_{G1}} &= \mathrm{RTT_1}\text{-}\left(\frac{\mathrm{W}}{2}+\frac{1}{\mathrm{r}}\right)\times\frac{\mathrm{M}}{\mathrm{D}} \quad [\mathrm{s}],\\ \mathrm{T_{Gn}} &= \mathrm{RTT_n}\text{-}\left(\frac{\mathrm{W}}{2}+\frac{\mathrm{n}}{\mathrm{r}}\right)\times\frac{\mathrm{M}}{\mathrm{D}} \quad [\mathrm{s}],\end{aligned} \tag{8}$$

where W is the maximum achievable congestion window given by Equation (6), M is the number of bits in a packet and D is the data rate at the sender node in bits per seconds.

The network node consumes peak power during packet transmission time and idle power when it is waiting for acknowledgements. We can model energy consumption of TCP by associating power levels to both phases of each transmission round. The energy consumption during a single RTT is formulated by following expression. A similar model is proposed in [29], even though such paper focuses only on the average power consumption of TCP and does not provide a deep insight of power consumption during each RTT.

$$\mathrm{En} = \mathrm{T_{G_n}}\times\mathrm{P_{Idle}} + \mathrm{T_{Tx_n}}\times\mathrm{P_{Peak}}[\mathrm{J}], \tag{9}$$

where n is RTT number, $\mathrm{P_{Idle}}$ and $\mathrm{P_{Peak}}$ are power consumption of a network node in Idle and transmitting modes and T_{Txn} is Time spent in transmitting packets for any RTT.

To find energy consumption for the complete TCP flow at the sender node we first estimate the energy consumption of one round by estimating the time spent in Gaps (T_{Ground}) and in transmitting packets (T_{Ground}). The energy consumption of TCP is same on sender, intermediate and receiver node for a single TCP flow.

$$\begin{aligned}T_{\mathrm{G_{round}}} &= \sum_{\mathrm{n}=0}^{\frac{\mathrm{rw}}{2}}\left(\mathrm{RTT}\text{-}\left(\frac{\mathrm{W}}{2}+\frac{\mathrm{n}}{\mathrm{r}}\right)\times\frac{\mathrm{M}}{\mathrm{D}}\right) =\\ &= \mathrm{RTT}\left(\frac{\mathrm{rW}}{2}+1\right)\text{-}\frac{\mathrm{M}}{\mathrm{D}}\left(\frac{3\mathrm{rW}^2}{8}+\frac{3\mathrm{W}}{4}\right) \quad [\mathrm{s}],\end{aligned} \tag{10}$$

Similarly, the time spent for transmitting packets in one round

$$T_{Tx_{round}} = \sum_{n=0}^{\frac{rW}{2}}\left(\frac{W}{2}+\frac{n}{r}\right)\frac{M}{D} = \frac{M}{D}\left(\frac{3rW^2}{8}+\frac{3W}{4}\right) \quad [s], \tag{11}$$

The energy consumption of TCP flow in single round is given by Equation (9) replacing T_{G_n} and $\mathrm{T_{Tx_n}}$ by (T_{Ground}) and $\mathrm{T_{Txround}}$. Equation (9) is used to measure energy consumption of TCP flow at a network node.

B. Solutions to improve energy efficiency

According to the TCP energy consumption model presented in previous section, the gaps between packet bursts and their size define the energy footprint of the TCP protocol on the network devices along the TCP data path. Each RTT, TCP sends packets in bursts and then waits for positive acknowledgements from the receiver.

DOI: 10.1002/dac

The energy consumption of a network node as well as the sender depends on their operation phase according to (9): it can reach the peak level during data transmission or just a fraction of it while the device is idle.

At the beginning of each round, the congestion window is usually around one-half of the end-to-end Bandwidth-Delay Product (BDP), and TCP spends considerable amount of time waiting for acknowledgements from the receiver after sending each burst of packets. The size of such waiting periods reduces as congestion window evolves and approaches BDP. After the congestion window exceeds the available network bandwidth, the packets begin to queue in the buffer and then, when the buffer is full, start to be dropped—triggering window reduction at the sender node and representing the potential beginning of a new round.

TCP spends a considerable amount of time in idle intervals (i.e. when it is not transmitting any packets), but network nodes and network interface cards remain operational and potentially waste energy. Such idle intervals represent a great potential for energy saving. In the following, we present three solutions to reduce energy consumption of TCP, which can be implemented both at the sender and in transit nodes.

2.1. Sleep flag in TCP header

Network devices require a fixed time for the transition to sleep mode or to wake up. In addition, they generally require an explicit control signal to put them to sleep. For this, to allow network devices (network routers and network interface cards) save power during protocol inactivity periods, an explicit signaling mechanism can be employed. TCP sender node can include a sleep flag into the optional field of TCP header by clearly marking the last packet in burst and indicate the estimated duration of inactivity period in milliseconds (see Figure 2). This flag marks the beginning of protocol inactivity period—during which network nodes can activate low power modes to conserve energy until the next burst of packets is received. The estimation of inactivity period is based on the TCP estimation of round trip times with timestamps, which might not be precise, but accurate enough to understanding whether it is worth initiating sleep mode or if the expected inactivity interval is too small. For each round of TCP flow, the inactivity period is estimated using Equation (8) and added in the options field of TCP header. Options kind field contains the sleep flag, options length indicates the total length of options field including options kind field and options data field contains the estimated inactivity period (see Figure 3). Gupta *et al*. estimated transitioning time of a network interface under 0.5 ms [30].

Energy consumption of a network device increases proportionally with round trip time. British Telecommunications summarized the average round trip time for intra Europe and inter continent traffic on a month scale of year 2014 [31]. Intra Europe is quite consistent with 27-ms RTT. Europe–Asia traffic experiences highest values of RTTs with 320 ms in June 2014. In similar fashion, yearly statistic report produced by Verizon Enterprise shows a quite variable behavior of RTT around the globe. The RTT can go up to 405 ms for traffic between UK and Australia [32]. In

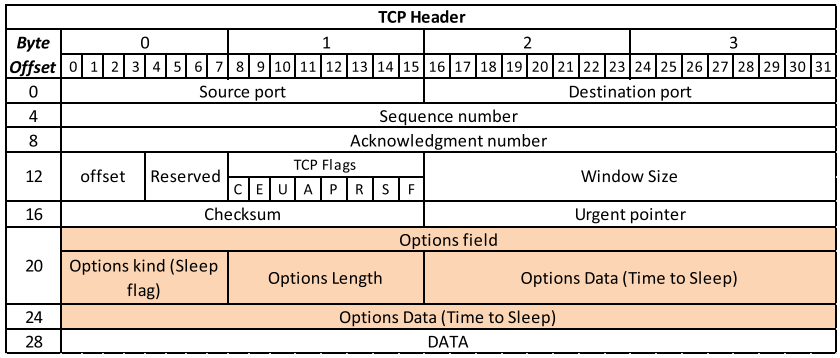

Figure 2. Sleep flag in TCP header.

DOI: 10.1002/dac

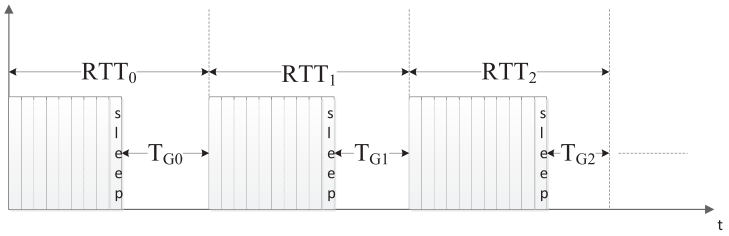

Figure 3. TCP flow with "sleep flags".

general RTTs in metropolitan areas are in the order of 30–50 ms, on the transatlantic connections 150–200 ms, while those involving satellite links are greater than 200 ms [33].

The percentage of mobile web traffic has increased rapidly in recent years. In 2018, according to Cisco visual networking index, in Western Europe mobile data traffic will share 83% of the total internet traffic of the region [34]. The mobile web gives comparatively higher values of round trip time. Users experience average round trip time of 400 ms on the sprint 3G network and 200 ms on sprint 4G network [35]. These high values of RTTs result in a waste of energy due to the inactivity intervals of TCP. Putting network interface of mobile/intermediate devices to sleep during these idle intervals can save a significant amount of energy.

2.2. Reducing inter packet gaps

TCP sender tends to produce packets in bursts, a behavior that represents a key for efficiency of many energy saving techniques. However, the packets transmitted head-to-back at the sender can increase inter packet gaps due to bottleneck links on the network path. The bottleneck link creates dispersion between packets due to its slow processing capacity [36]. The dispersion between packets is created with buffering packets at bottleneck link while waiting for the output link to be available. The subsequent link forwards this traffic at higher speed creating time gaps between packets. In the same fashion, acknowledgements arrive at the source with the same time gaps. TCP starts sending next congestion window upon receiving the first acknowledgment from receiver. Every received acknowledgment results in the transmission of one data packet. Therefore, time gaps between acknowledgments generate similar gaps between data packets when transmitted from source.

Compressing these inter packet gaps can save a significant amount of energy. The transition time of a network device (or interface) required for transitioning to sleep mode or wake up is almost 0.5 ms [30]. This 0.5 ms includes both the transition time as well as synchronization of links when links need to be re-established. We observed that the inter-packets gap can go above 1 ms. As a consequence, we propose a solution to combine inter packet gaps by buffering packets and sending them in small bursts. As an example, if the inter-packet gap for a TCP flow is 1 ms, then buffering 16 packets creates a time gap of 15 ms that is well above the transition time of network device. The packets are sent in burst afterwards. We buffer acknowledgments at node 2 instead of data packets at node 1. Buffering acknowledgments prevents their processing at node 1, and it can be put to sleep while acknowledgments are being buffered at node 2. In Figure 4 packet transmission diagram of two links is shown with inter packet gaps on 100-Mbps links. This buffering of acknowledgments (Figure 5) combines packet inactivity period during TCP flow, and network device is put to sleep to save energy. In Section III the impact of buffering on TCP throughput is estimated with estimation of energy saving on network devices.

2.3. Delayed acknowledgements

Delayed acknowledgements is a technique where receiving nodes combine several acknowledgement responses in a single acknowledgment packet. Delaying acknowledgements reduces the number of acknowledgements received hence leads to a slower increase of congestion window both in slow start and congestion avoidance phase. Thus, the process of congestion window evolution is slowed down as given in equation (1). The congestion window is increased with $1/r$ with each RTT. This causes TCP to spend more time in inactivity period in the start of each round due to slow evolution of congestion

DOI: 10.1002/dac

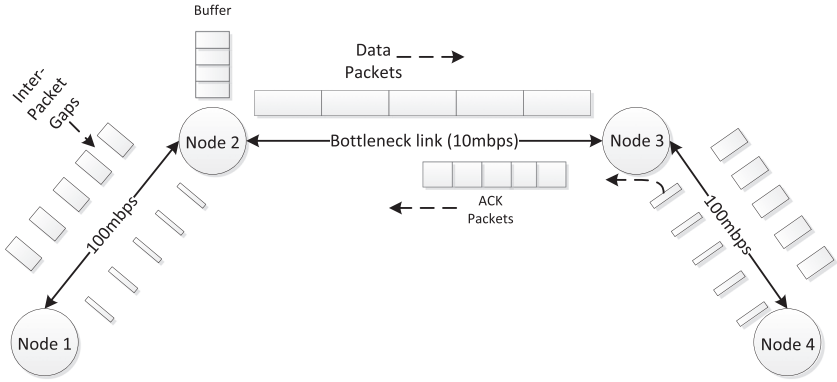

Figure 4. Inter packet gaps due to bottleneck links.

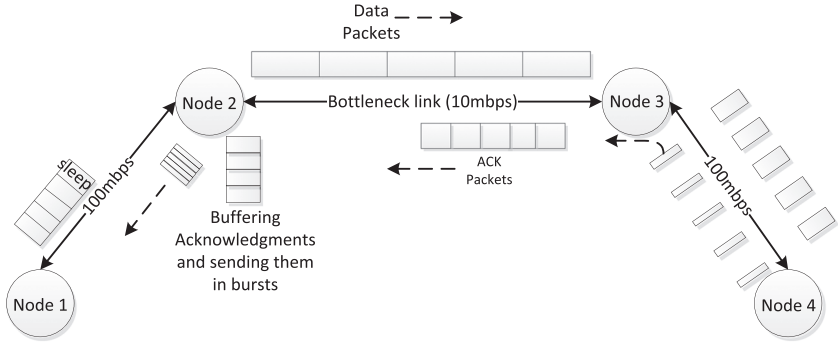

Figure 5. Buffering acknowledgments at node 2 to combine inter packet gaps.

window. For $r=2$ the congestion window remains same for consecutive round trip times and then increases by 1 for next two RTTs as shown in Figure 6. In this way, we have more time to put network nodes to sleep and save energy in the start of each round.

Basically the delayed acknowledgement procedure defines two parameters: delayed ACK number and delayed ACK timer. The delayed ACK number defines the number of packets for which the receiver waits before releasing an acknowledgment packet. Studies have shown that $r=2$ gives the optimal performance in terms of throughput [37]. While second parameter delayed ACK timer is set by TCP, depending on which delayed acknowledgement procedure is modified. Acknowledgement is sent if the timer is expired. Figure 7 explains the role of acknowledgement timer (100 ms) where acknowledgments are sent at the event of timer expiration.

We performed different experiments for different values of delayed acknowledgement coefficient and see the impact on energy consumption of TCP and throughput. The graphs showing energy consumption of different flows with different values of r (delaying acknowledgement index) are shown in evaluation section.

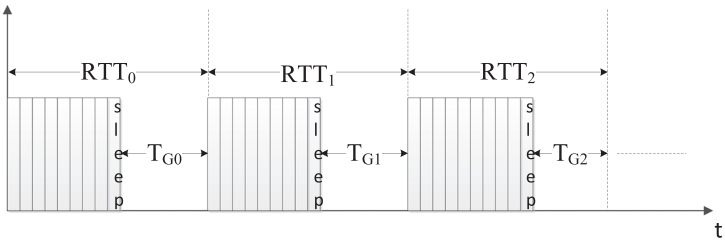

Figure 6. TCP congestion window evolution with delayed acknowledgment, $r=2$.

DOI: 10.1002/dac

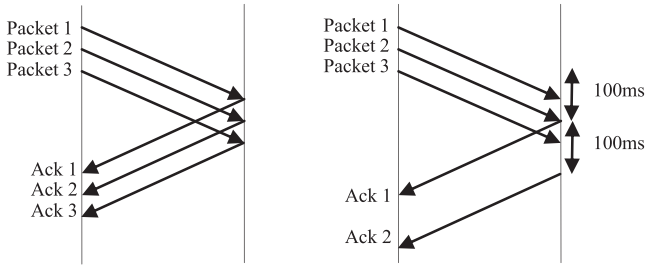

Figure 7. Role of delayed acknowledgements in TCP.

3. EVALUATION

This section presents performance evaluation of the proposed solutions for improving energy efficiency of TCP by presenting numerical results obtained mainly from the model introduced in Section II and then the simulation results obtained during Ns2 simulations [38].

A. Numerical results

Figure 8 presents the evolution of energy consumption for different RTT values obtained using Equation (9) for a 27-MB file. The observed energy consumption of network devices is proportional to RTT. For RTTs smaller than 50 ms, which is typical for metropolitan-area networks, the consumed energy ranges from 0.9 K To 42 K joules. For the RTTs of 50–200 ms, typical for transatlantic connections, the consumed energy ranges from 42 k to 166 K joules. Finally, for the RTTs of 200–300 ms, typical for satellite links, the network devices consume more than 250 K joules of energy. Larger RTTs slow down TCP window evolution and increase time the TCP sender remains silent waiting for the acknowledgements from the receiver. Reducing these intervals of inactivity or placing sending devices along the path into sleep would save energy and has a potential to make energy consumption independent of RTT.

For all numerical and simulation results network nodes use the following values of power consumption: 300 W for peak consumption, 212 W for idle mode and 1 W during the sleep state. These values are consistent with Cisco Catalyst 4948 10 Gigabit Ethernet switch commonly used for Top-of-Rack (ToR) switches in modern data centers [39].

Figure 8 presents the effect of adding sleep flag in TCP header on the energy consumption of network node. The network node is put to sleep if packet inactivity period is longer than 1 ms, which corresponds to the time network switches take to enter the sleep mode and wake up [30]. Smaller

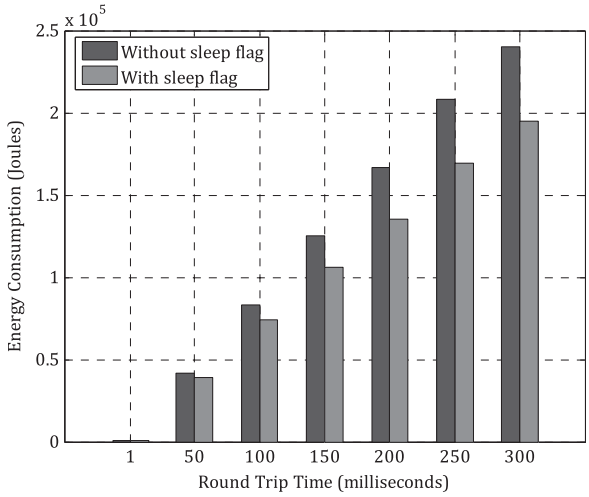

Figure 8. Energy consumption of network nodes with and without sleep flag in TCP header.

DOI: 10.1002/dac

inactivity periods, which are typical for LANs, do not provide substantially enough time for transitioning into the power saving mode.

For the RTTs in the order of 50 ms, individual network nodes consume 47.3 K joules energy each during normal operation and 44.8 K joules with sleep mode enabled during idle periods. This saves around 6% energy for a single flow. The TCP throughput is not affected, as an addition of a sleeping flag does not alter protocol basic principles and behavior. For larger RTTs of 300 ms, the energy savings are higher and in the order of 19%.

The transport layer of TCP sender node adds the sleep flag and includes calculation of the expected inactivity period using Equation (8). Having received this flag, the receiver as well as network routers located on the end-to-end path use indicated protocol inactivity period to guide their sleep patterns.

B. *Simulation results*

Figure 9 details the considered simulation scenario with a bottleneck link. The bottleneck link buffer is limited by 50 packets. During simulations, the TCP sender transmits a file of 27 MB to the receiver, which replies with acknowledgements and silently discards all the received data.

3.1. Sleep flag in TCP header

Figure 10 presents the energy consumption of network devices obtained using simulations for scenarios with and without the sleep flag. As expected, longer RTTs provide more room for energy saving. For RTTs equal to 50 ms, the saving is around 5%, for RTTs equal to 250 ms 17% and up to 20% can be saved for RTTs of 300 ms. It is important to note, that the throughput remains

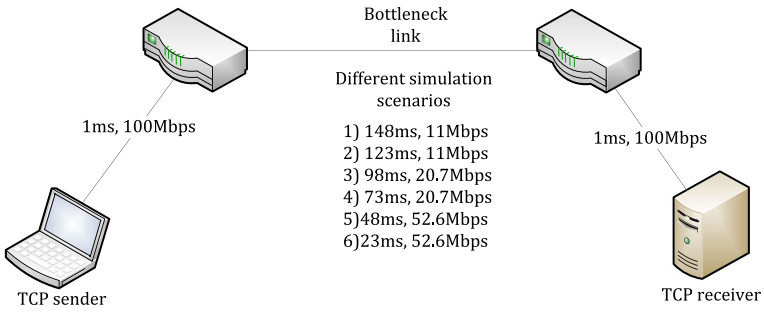

Figure 9. Simulation scenario.

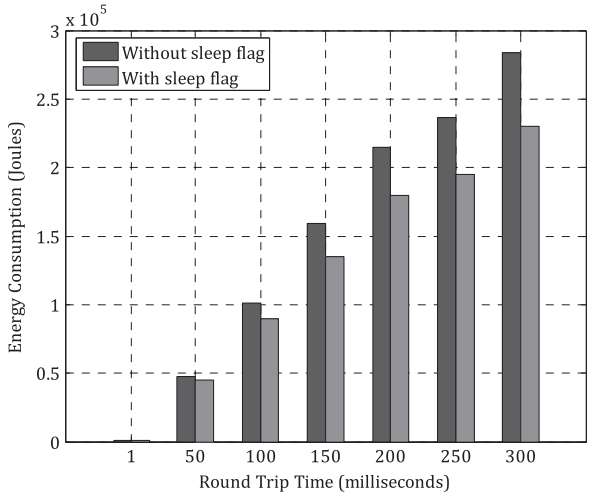

Figure 10. Energy consumption of network nodes with sleep flag in TCP header.

DOI: 10.1002/dac

the same independently of using the sleep flag option in TCP header, as it affects only protocol inactivity periods.

3.2. Reducing inter-packet gaps

In simulation scenario, the bottleneck link has a capacity of 11 Mbps, which is almost ten times smaller than 100-Mbps link connecting the TCP sender node. Due to this bandwidth mismatch, the bulk of TCP packets transmitted head-to-back becomes distorted and inter-packet gaps are inserted. In addition, the acknowledgements generated at the receiver will encapsulate these inter-packet gaps and propagate them back to the sender (see Figure 4).

These inter-packets gaps are periods of inactivity in TCP congestion control algorithm, during which network nodes remain idle, but still consume significant amount of energy. The size of inter-packet gaps is usually in the order of milliseconds, which is comparable with the time required by the network node to go to sleep and then wake up. This makes it not feasible to perform dynamic shutdown for each individual gap and requires first to make the gaps grouped together.

Grouping inter-packet gaps can be done by buffering packets (see Figure 5). The receiver buffers acknowledgments before sending them back to the sender. Figure 11 shows simulation results, which confirm that amount of the saved energy is proportional to RTT size. Up to 83.67% of energy can be saved for RTTs of 300 ms, if up to 16 acknowledgement packets can be buffered. As depicted in Figure 11, a TCP flow consuming 284 K joules of energy starts to consume only 43.8 K joules for same TCP flow. Energy saving is more effective for longer RTTs. The technique of reducing inter-packet gaps can be used together with the sleep flag in TCP header in order to explicitly notify network nodes of the beginning of combined idle periods.

Figure 12 illustrates comparison of throughput obtained with reduced inter-packet gaps and standards TCP flow. For RTTs equal to 300 ms, the average throughput is 9.09 Mbps for standard TCP flow and 8.78 Mbps with reduced gaps and sleep flag enabled in TCP header. This corresponds 3.4% degradation in throughput, while the energy saving achieved is of 83.67%. The average throughput degradation ranges from 3.44% to 10.04%, while at the same time the average range of energy saving is 10.35% to 83.67% with reduced gaps and sleep flag enabled in TCP header.

3.3. Delayed acknowledgments

The technique of confirming the successful reception of several data packets with a single acknowledgement, called delay acknowledgements, is known to improve the throughput of TCP, but also appears to be a good tool for reducing energy consumption of TCP.

Figure 13 presents energy consumption of TCP flow for different values of the delayed acknowledgment coefficient *r*. It is observed that, TCP consumes maximum energy if every

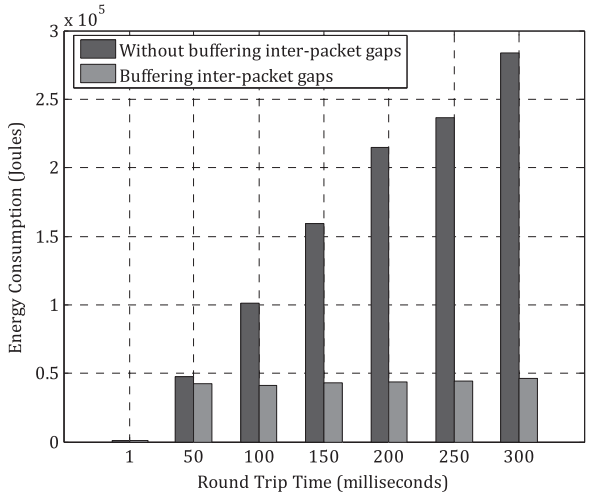

Figure 11. Energy consumption of TCP with reduced inter-packet gaps.

DOI: 10.1002/dac

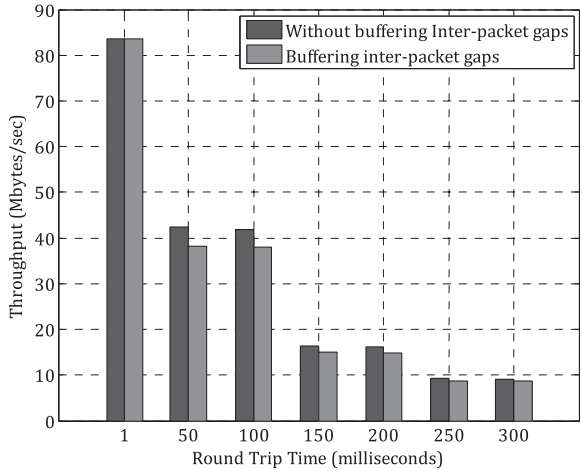

Figure 12. TCP throughput with reduced inter-packet gaps.

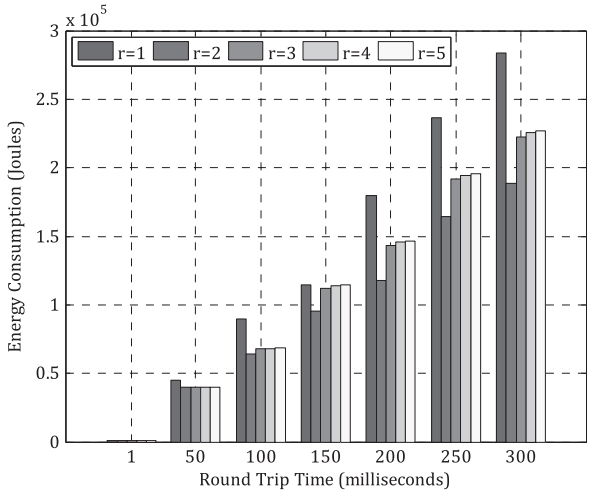

Figure 13. Energy consumption with delayed acknowledgements.

packet is acknowledged. The energy consumption decreases with delaying TCP acknowledgements, and the least energy is consumed if every other packet is acknowledged. Table I shows the percentages of energy saving for different values of RTT with different delayed acknowledgement coefficients r. The average range of energy saving is 0.89% to 33.43% with delayed acknowledgement coefficient $r=2$.

Table I. Energy saving with delayed acknowledgements.

	Energy saving delayed acknowledgement coefficient			
RTT	$r=2$	$r=3$	$r=4$	$r=5$
1 ms	0.89%	0.65%	0%	0%
50 ms	12.56%	11.43%	11.30%	11.42%
100 ms	28.67%	24.43%	23.91%	23.67%
150 ms	16.85%	2.45%	0.69%	0.33%
200 ms	31.47%	20.38%	18.87%	18.65%
250 ms	30.50%	18.90%	17.83%	17.08%
300 ms	33.43%	21.62%	20.52%	19.90%

DOI: 10.1002/dac

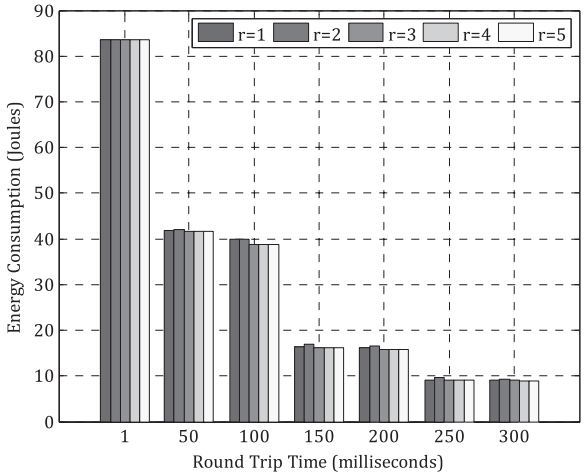

Figure 14. Impact of delayed acknowledgment on TCP throughput.

Delaying TCP acknowledgments decreases energy consumption of network nodes. Delayed acknowledgements slow down congestion window evolution making TCP to spend more time in inactivity periods. Simulation results show that the highest savings and throughput are achieved with having one acknowledgement for every two data packets on average, which corresponds to the case of $r=2$.

Almost similar trend is observed for TCP throughput with small variation for different delaying acknowledgement coefficients (Figure 14). $R=2$ gives optimal performance for all RTTs with little difference as compared to other values of r.

4. CONCLUSION AND FUTURE WORK

This paper presents a mathematical model to estimate the energy consumption of TCP and three novel solutions, which can be used to improve energy efficiency of network nodes. The solutions, which include adding sleep indication flag in the TCP header, reduction of inter-packet gaps and employing delayed acknowledgements, take advantage of the burstiveness nature of TCP streams and enable power saving during inactivity periods of the protocol. The achieved savings can be as high as 80% for a single TCP flow.

The future work will focus on extending the proposed model to multi-flows scenarios and network nodes able to alter line speed to conserve energy. Finally, application of cognitive techniques [40] will be explored to address the impact of energy efficiency in cloud computing data centers [41].

ACKNOWLEDGEMENTS

The authors would like to acknowledge the funding from National Research Fund, Luxembourg in the framework of ECO-CLOUD project (C12/IS/3977641).

REFERENCES

1. Greening the digital world. http://ec.europa.eu/commission_2010-2014/kroes/en/blog/ict-footprint
2. Sustainable ICT in corporate organizations. http://www.itu.int/dms_pub/itu-t/oth/4B/04/T4B0400000B0011PDFE.pdf
3. Gunaratne C, Christensen K, Nordman B, Suen S. Reducing the energy consumption of Ethernet with Adaptive Link Rate (ALR). *IEEE Transactions on Computers* 2008; **57**(4): 448–461. DOI:10.1109/TC.2007.70836.
4. Pamlin D, Szomolanyi K. Saving the climate @ the speed of light. First roadmap for reduced CO_2 emissions in the EU and beyond, World Wildlife Fund and European Telecommunications Network Operators 2007.
5. Bianzino AP, Chaudet C, Rossi D, Rougier JL. A survey of green networking research. *IEEE Communication surveys & tutorials* 2012; **14**(1): 3–20. DOI:10.1109/SURV.2011.113010.00106.

6. Gartner says data centers account for 23 per cent of global ICT CO2 emissions, Press release, 11 October 2007. http://www.gartner.com/it/page.jsp?id=530912
7. Fan X, Weber WD, Barroso LA. Power provisioning for a warehouse-sized computer. *SIGARCH Comput. Archit. News* 2007; **35**(2): 13–23. DOI:10.1145/1273440.125066.
8. Raghavendra R, Ranganathan P, Talwar V, Wang Z, Zhu X. No "power" struggles: coordinated multi-level power management for the data center. *ACM SIGOPS Operating Systems Review* 2008; **42**(2): 48–59. DOI:10.1145/1353535.1346289.
9. GeSI, "Smart 2020: enabling the low carbon economy in the information age", (2008). http://www.smart2020.org/_assets/files/02_smart2020Report.pdf
10. Murugesan S, Gangadharan GR. Harnessing Green IT: Principles and Practices, John Wiley & Sons Ltd 2013; DOI: 10.1002/9781118305393.ch7.
11. Antonopoulos A, Verikoukis C. Multi-player Game Theoretic MAC Strategies for Energy Efficient Data Dissemination. *IEEE Transactions on Wireless Communications* 2014; **13**(2):592–603, DOI: 10.1109/TWC.2013.120713.120790.
12. Antonopoulos A, Verikoukis C, Skianis C, Akan OB. Energy Efficient Network Coding-based MAC for Cooperative ARQ Wireless Networks. *Ad Hoc Networks* 2013; **11**(1): 190–200, DOI: 10.1016/j.adhoc.2012.05.003.
13. Antonopoulos A, Di Renzo M, Verikoukis C. Effect of Realistic Channel Conditions on the Energy Efficiency of Network Coding-aided Cooperative MAC Protocols. *IEEE Wireless Communications Magazine* 2013; **20**(5): 76–84, DOI: 10.1109/MWC.2013.6664477.
14. Bilal K, Khan SU, Madani SA, Hayat K, Khan MI, Min-Allah N, Kolodziej J, Wang L, Zeadally S, Chen D. A survey on Green communications using Adaptive Link Rate. *Springer Cluster Computing* 2013; **16**(3): 575–589. DOI:10.1007/s10586-012-0225-8.
15. Irish L, Christensen KJ. A "Green TCP/IP" to reduce electricity consumed by computers. in *Proceedings of the IEEE Southeastcon* 1998; 302–305, DOI: 10.1109/SECON.1998.673356.
16. Chabarek J, Banerjee S, Sharma P, Mudigonda J, Barford P. Networks of Tiny Switches (NoTS): In search of network power efficiency and proportionality, in Proceedings of the 5th Workshop on Energy-Efficient Design 2013.
17. Hagag S, El-Sayed A. Enhanced TCP Westwood congestion avoidance mechanism (TCP WestwoodNew). *International Journal of Computer Applications* 2012; **45**(5): 21–29. DOI:10.5120/6776-9071.
18. Sandvine Intelligent Broadband Networks, Global internet phenomena report 2013. https://www.sandvine.com/downloads/general/global-internet-phenomena/2013/sandvine-global-internet-phenomena-report-1h-2013.pdf
19. Wang B, Singh S. Computational energy cost of TCP," in Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies 2004; 2:785–795, DOI 10.1109/INFCOM.2004.1356967.
20. He C, Sheng B, Zhu P, Wang D, You X. Energy efficiency comparison between distributed and co-located MIMO systems. *International Journal of Communication Systems* 2014; **27**(1): 81–94. DOI:10.1002/dac.2345.
21. Nordma B. EEE savings estimate 2007. http://grouper.ieee.org/groups/802/3/eee_study/public/may07/nordman_2_0507.pdf
22. Patel-Predd P. Energy-efficient Ethernet. *IEEE Spectrum* 2008; **45**(5): 13. DOI:10.1109/MSPEC.2008.4505297.
23. Zhang B, Sabhanatarajan K, Gordon-Ross A, George A. Real-time performance analysis of adaptive link rate, 33rd IEEE Conference on Local Computer Networks (LCN) 2008; 282–288, DOI: 10.1109/LCN.2008.4664181.
24. Stevens W. *TCP/IP Illustrated (Vol 1): The Protocols*, Addison-Wesley, New York 1994; ISBN:0-201-63346-9.
25. Hu Z, Li G. On energy-efficient TCP traffic over wireless cooperative relaying networks. *EURASIP Journal on Wireless Communications and Networking* 2012; **2012**(78), DOI: 10.1186/1687-1499-2012-78.
26. Ayadi A, Maille P, Sanchez DR. TCP over low-power and lossy networks: tuning the segment size to minimize energy consumption, 4th IFIP International Conference on New Technologies. *Mobility and Security (NTMS) Paris* 2011; **1–5**: . DOI:10.1109/NTMS.2011.5720608.
27. Hashimoto M, Hasegawa G, Murata M. An analysis of energy consumption for TCP data transfer with burst transmission over a wireless LAN. *International Journal of Communication Systems* 2014. DOI:10.1002/dac.2832.
28. Mathis M, Semke J, Mahdavi J. The macroscopic behavior of the TCP congestion avoidance algorithm. *ACM Computer Communicatons Review* 1997; **27**(3): 67–82. DOI:10.1145/263932.264023.
29. Singh H, Singh S. Energy consumption of TCP Reno, Newreno, and SACK in multi-hop wireless networks. *ACM SIGMETRICS Performance Evaluation Review—Measurement and modeling of computer systems* 2002; **30**(1): 206–216. DOI:10.1145/511399.511360.
30. Gupta M, Singh S. Dynamic ethernet link shutdown for energy conservation on ethernet links. *IEEE International Conference on Communications (ICC) Glasgow* 2007; **6156–6161**. DOI:10.1109/ICC.2007.1019.
31. British Telecommunications—Monthly Network Summary. http://ippm.bt.net/month.shtml
32. Verizon Enterprise Solutions—Latency Statistics. http://www.verizonenterprise.com/about/network/latency/
33. Verizon, Global Latency and Packet Delivery SLA. http://www.verizonenterprise.com/terms/global_latency_sla.xml
34. Cisco visual networking index: global mobile data traffic forecast update, 2013–2018. http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html
35. Latency: the new web performance bottleneck. https://www.igvita.com/2012/07/19/latency-the-new-web-performance-bottleneck/
36. Kapoor R, Chen L, Lao L, Sanadidi M, Gerla M. CapProbe: a simple and accurate capacity estimation technique. *ACM SIGCOMM Computer Communication Review* 2004; **34**(4): 67–78. DOI:10.1145/1030194.1015476.
37. Puri P, Kumar G, Tripathi B, Kaur G. Analysis of DelAck based TCP-NewReno with varying window size over Mobile Ad Hoc Networks International. *Journal of Computer Science and Information Security* 2012; **10**(1):62–67.

38. The Network Simulator—ns2. http://www.isi.edu/nsnam/ns/
39. Cisco Catalyst 4948 10 Gigabit Ethernet switch. http://www.cisco.com/c/en/us/products/collateral/switches/catalyst-4900-series-switches/product_data_sheet0900aecd80246552.html
40. Granelli F, Kliazovich D, Malheiros N. Towards cognitive internet: an evolutionary vision, Springer. *Cognitive Radio and Networking for Heterogeneous Wireless Networks* 2014; 181–200. DOI:10.1007/978-3-319-01718-1_6.
41. Kliazovich D, Bouvry P, Granelli F, Fonseca N. *Energy Consumption Optimization in Cloud Data Centers*, Wiley: Cloud Services, Networking and Management, 2015.

DOI: 10.1002/dac