# Modeling The Interaction between TCP and Rate Adaptation at Links

Faheem Khuhawar, Marco Mellia, Michela Meo
Dip. di Elettronica e Telecomunicazioni, Politecnico di Torino, Italy.
Email: {firstname.lastname}@polito.it

***Abstract*—In this paper, we model and investigate the interaction between the TCP protocol and rate adaptation at intermediate routers. Rate adaptation aims at saving energy by controlling the offered capacity of links and adapting it to the amount of traffic. However, when TCP is used at the transport layer, the control loop of rate adaptation and one of the TCP congestion control mechanism might interact and disturb each other, compromising throughput and Quality of Service (QoS). Our investigation is lead through mathematical modeling consisting in depicting the behavior of TCP and of rate adaption through a set of Delay Differential Equations (DDEs). The model is validated against simulation results and it is shown to be accurate.**

**The results of the sensitivity analysis of the system performance to control parameters show that rate adaptation can be effective but a careful parameter setting is needed to avoid undesired disruptive interaction among controllers at different levels, that impair QoS.**

## I. Introduction

In last decade, Information and Communication Technology (ICT) has been observed as an evident energy consumer and number of initiatives are taken to reduce energy consumption and improve energy efficiency [1]. Furthermore, the rising trend of the amount of traffic, due to the diffusion of video and data services, as well as the more and more widespread use of communication devices in general, and dramatic improvements in technologies to make them more efficient have severely increased the energy demand. Hence, in all fields of ICT and for all technologies, robust countermeasures are required to cope with the rapid growing trend of energy consumption.

In the context of wired networks, recently the interest in saving energy consumption has brought the emergence of new proposals and standards. For instance, the IEEE Energy Efficient Ethernet 802.3az standard aims at improving efficiency of the traditional Ethernet protocol, that requires that the transmitter and the receiver remain continuously active even if there is no data to send, leading to waste of energy (about 0.5W for 1GBase-T and 5W for 10GBase-T), by entering low consuming sleep modes when there is no traffic. The adoption of the new standard allows to save an amount of energy that, in the best case, i.e., when the transmission consists of large bursts and gaps, becomes roughly proportional to link utilization; however, limited advantages are achieved when the transmission consists of small packets with relatively small and regular inter-packet gaps. Thus, some works try to overcome this problem, and aim at more efficient energy proportionality, as [2] proposes to send cumulative Acknowledgements (ACKs), i.e., use delay parameter to send ACKs together after certain time or use larger frame size. In [3], it is proposed to send packets as a burst by collecting the packets (packet coalescing) in queue with a certain limit and send them at once on the link.

Another quite promising approach to save energy and to make the devices energy consumption more proportional to the offered load, consists in *rate adaptation*. Leveraging on the fact that less energy is needed when low transmission rates are used instead of high rates, the idea consists in adjusting the data transmission rate or capacity according to traffic dynamics: when traffic is high, the maximum capacity is used, but when traffic is low, low transmission rates, that consume less energy, are used. For instance, [4] proposes to use discrete transmission rates according to traffic intensity. While [5] takes another approach, that is to adapt the data rate according to link utilization. In this paper, we focus on this kind of approach. Thus, the objective of our rate controller is to maximize the energy savings without compromising the QoS.

The rate adaptation mechanisms implemented in the router might, however, negatively interact with other mechanisms implemented at upper levels. In particular, we consider here the possible interaction between the rate adaptation control loop and TCP congestion control loop [6]. The scenario is quite important, being TCP the transport protocol that carries most of the traffic. The control loop defined in the congestion avoidance phase of TCP basically, i) increases the amount of traffic injected in the network when there seems to be enough bandwidth and, ii) performs multiplicative decrease of the amount of injected traffic upon loss indication. On its side, the rate adaptation scheme has a control loop that is controlled by the amount of traffic injected in the network, as seen by the queue size. Hence, the data rates are adapted based on the offered load. Thus, if not properly tuned, the two control loops might negatively interact. For example, a rate reduction of the rate adaptation algorithm might induce TCP to reduce injected traffic; in its turn, this TCP traffic reduction indicates that load decreases and might trigger further rate reductions in the router. This interaction might lead, finally, to bad performance and impair end-users Quality of Experience (QoE). Aim of this work is to know how these two control loops (rate adaptation loop and congestion control loop) interplay with each other.

To investigate the interplay of the two control loops, we propose in this paper a fluid model analysis. Hence, a queue

TABLE I
PARAMETERS AND VARIABLES

| Description | Symbol | Value | Unit |
|---|---|---|---|
| TCP congestion window | $W$ | $0:150$ | packets |
| Queue length | $Q$ | $0:100$ | packets |
| Estimated queue length | $\bar{Q}$ | $0:100$ | packets |
| Capacity | $C$ | $100:1000$ | packets/sec |
| Initial capacity | $C_0$ | 200 | packets/sec |
| Capacity controller threshold | $Q_{oc}$ | $5:25$ | packets |
| Loss controller threshold | $Q_{ol}$ | 20 | packets |
| Propagation delay | $R_o$ | 0.02 | seconds |
| Delay of loss indication | $\tau$ | $0.03:0.04$ | seconds |
| Round Trip Time | $RTT$ | $R_o + Q/C$ | seconds |
| Control interval (model) | $\delta_m$ | $1/1000$ | seconds |
| Control interval (simulator) | $\delta_s$ | $1/10000$ | seconds |
| Loss scaling factor | $K_l$ | $1/80$ | – |
| Capacity scaling factor | $K_c$ | $1:5000$ | – |
| Smoothing factor | $\alpha$ | $0:1$ | – |
| Loss rate | $L$ | $0:1$ | – |

length driven rate adaptation scheme is considered and analyzed through mathematical modeling using a fluid model based framework. Several experiments are then conducted through simulations to validate the results obtained from the model.

The paper is organized as follows. In section II, we start by describing the model through the set of differential equations along with the two proposed rate adaptation models. Later, time evolution of parameters obtained by solving the system of differential equations along with its simulation results to validate the model are discussed in section III. Steady state analysis of the model and the impact of tuning the parameter values are highlighted in section IV. Lastly, comparison of results obtained from both the models plus conclusion drawn are mentioned.

## II. FLUID MODEL

The setting up of rate adaptation scheme requires the modeling of TCP behavior. Hence, a set of coupled Delay Differential Equations (DDEs) are derived to describe the basic TCP evolution. We build upon the differential equations proposed in [7] that depict the well known TCP congestion control mechanism. The solution of the model provides the evolution in time of TCP as well as network parameters such as window size, queue length and capacity. The framework uses the fluid based model, wherein the parameters evolve as a continuous process. Table I reports the list of variables used in this paper.

### A. TCP Model

The TCP model focuses on the evolution of the congestion window size, that we denote by $W(t)$. We consider only the congestion avoidance mechanism of TCP, since this is the one that mainly determines the performance of long-lived TCP flows and we neglect the slow start phase. Furthermore, since we consider, as typical, a low loss regime, the timeout behavior is also neglected. The evolution of the congestion window size in congestion avoidance basically consists of two parts; the additive increase by one segment of MSS (Maximum Segment Size) every RTT and multiplicative decrease in case of loss indication. In our case, losses of type Triple Duplicate (TD) ACKs are considered. The TCP sender receives the loss indication (TD ACKs) after roughly one RTT. This delay parameter is denoted in our model by the symbol $\tau$. Thus, a loss indication received at time $t$ means that a loss actually occurred in the router at time $t - \tau$.

Similar to [7], we consider a scenario in which a Random Early Detection (RED) queue management algorithm is used in the router, thus, losses occur according to the RED mechanism that depend on the queue length. Further description about RED queue management algorithm can be found in [8]. An extention of the model to other queue management schemes is left for future work. Let the loss rate at time $t$ be denoted by $L(t)$. Given TCP delay in detecting losses, at time $t$ TCP perceives a loss rate equal to $L(t-\tau)$. The number of losses is thus given by the product of the loss rate times the throughput, that, in its turn, is given by ratio between the congestion window size and the round trip time; the number of losses is thus equal to $L(t-\tau)\frac{W(t-\tau)}{RTT(t-\tau)}$.

Under the fluid model based framework, the RED queuing policy can be modeled by making losses proportional to queue length, according to the following,

$$L(t) = K_l(Q(t-\tau) - Q_{ol}) \;\; for \;\; 0 \leqslant L(t) \leqslant 1 \qquad (1)$$

where $L(t)$ represents the loss rate, and $K_l$ is the loss rate scaling factor that defines the growth of the drop probability function in the RED queue. $Q_{ol}$ is the loss threshold, thereafter the RED queue starts to drop packets. Similarly, $Q(t-\tau)$ represents the current queue length at time $t-\tau$, i.e., when the transmitter sees the loss.

The evolution of the round trip time takes the following form,

$$RTT(t) = R_o + \frac{Q(t)}{C(t)} \qquad (2)$$

where $R_o$ is the propagation delay (that is constant) and the fraction $\frac{Q(t)}{C(t)}$ corresponds to the queuing delay measured in seconds. Similarly, $RTT(t-\tau) = R_o + \frac{Q(t-\tau)}{C(t-\tau)}$.

Putting the above mechanisms together, the TCP congestion avoidance behavior can be modeled by the following DDE,

$$\frac{dW(t)}{dt} = \frac{1}{RTT(t)} - \frac{W(t)}{2}\left(L(t-\tau)\frac{W(t-\tau)}{RTT(t-\tau)}\right) \qquad (3)$$

where the derivative $\frac{dW(t)}{dt}$ represents the change of window size measured in packets.

The differential equation that describes the queue length evolution can be written as the difference between the arrival rate of packets at the queue, represented by the term $\frac{W(t)}{RTT(t)}$ and the rate at which the packets are served, represented by

the capacity $C(t)$,

$$\frac{dQ(t)}{dt} = \frac{W(t-\tau)}{RTT(t-\tau)} - C(t) \;\; for \;\; 0 \leqslant Q(t) \leqslant Q_{max} \quad (4)$$

The differential equation is bounded between 0 and $Q_{max}$, so that the serving of packet occurs when the queue length is greater than zero.

### B. Rate Adaptation

We describe here the model of the rate adaptation algorithm, i.e., of the strategy with which the transmission rate is adapted to the queue length. In particular, we focus on two slightly different controllers: in the first one, the rate is adapted to the *instantaneous queue length*, in the second one a *weighted average of the queue length* is used as control parameter.

*1) Rate Controller I:* The idea is to design a rate controller that adapts the transmission rate according to queue length evolution. In other words, the service rate or capacity $C(t)$ increases/decreases if the instantaneous queue length grows above/below a certain threshold $Q_{oc}$. Based on the value of threshold parameter, the controller has the ability to aggressively operate at maximum capacity (low value of $Q_{oc}$) i.e., minimal energy saving while maintaining high QoS; or aggressively reduce energy consumption at the expense of possibly compromising QoS (high value of $Q_{oc}$). The rate controller can be modeled by,

$$\frac{dC(t)}{dt} = K_c(Q(t) - Q_{oc}) \;\; for \;\; C_{min} \leqslant C(t) \leqslant C_{max} \quad (5)$$

where $K_c$ represents the capacity scaling factor and $Q_{oc}$ the queue length threshold, with the rate controller being bounded between minimum and maximum capacity constraints.

*2) Rate Controller II:* In this case, the rate controller adapts the rate to an average queue length, and, in particular to the Exponentially Weighted Moving Average (EWMA) queue length instead of the instantaneous queue length. The purpose is to possibly make the controller more stable by adapting the rate more slowly to queue length variations. The EWMA smooths the variations of the queue length through a smoothing factor, $\alpha$, with $0 < \alpha < 1$. A queue length sampling interval equal to $\delta_m$ is used. The value of $\delta_m$ is chosen to be fixed and it is set to $1/C_{max}$, which means the value is updated on almost every packet transmission. Hence, the equation describing the evolution of EWMA queue length as shown in [7] is as follows,

$$\frac{d\bar{Q}(t)}{dt} = \frac{\log(1-\alpha)}{\delta_m}\bar{Q}(t) - \frac{\log(1-\alpha)}{\delta_m}Q(t) \quad (6)$$

where $\bar{Q}(t)$ represents the average queue length. Therefore, the rate controller based on estimated queue length can be described with the following equation,

$$\frac{dC(t)}{dt} = K_c(\bar{Q}(t) - Q_{oc}) \;\; for \;\; C_{min} \leqslant C(t) \leqslant C_{max} \quad (7)$$

## III. Model Validation

In this section we first describe the kind of results that can be obtained by the model and provide a first analysis of them; we then validate the analytical model by comparison against simulation results.

### A. Time Evaluation

The set of coupled delay differential equations (3), (4), (5), i.e., the system of differential equations, are solved numerically in order to get the evolution in time of the mentioned parameters. To derive a first set of results, we set the model parameters as follows. The initial starting capacity represented with $C_0$ to 200 $^{\text{packets}}/_{\text{sec}}$; operating within the range of $C_{min} = 100$ to $C_{max} = 1000$ $^{\text{packets}}/_{\text{sec}}$. The capacity threshold parameter is set to $Q_{oc} = 5$ packets. Similarly, the loss threshold is fixed to $Q_{ol} = 20$ packets. The queue length is bounded in the region of $Q_{min} = 0$ to $Q_{max} = 100$ packets. The loss scaling factor $K_l = 1/80$, which is equivalent to RED drop bound going from 20 to 100 packets with packet discard probability function of value 1, when the queue length is equal to the maximum, $Q_{max}$. The propagation delay is fixed to $R_o = 0.021$ seconds and the loss delay indication factor to $\tau = 0.036$ seconds.

Figure 1 illustrates an example of the evolution of the parameters based on rate controller I. Observe the fact that the capacity, after some transient period, achieves, as desired, the maximum possible value. The congestion window size, as well as the queue length, continuously vary in time and it oscillates, as is expected from long-lived TCP flows in congestion avoidance. The evolution of parameters based on rate controller II is pretty much the same for low values of $K_c$, whereas the evolution of capacity is much smoother for higher values of $K_c$ with respect to rate controller I, see Section IV for detailed discussion.

### B. Simulation Results

*1) Design Parameters:* To validate the rate adaptation models, a simple network topology is simulated through a discrete event simulator developed in OMNeT++. The considered scenario is shown in Fig. 2, consists of two nodes, namely client and server, that communicate with each other using the TCP protocol (TCP Reno flavor along with default TCP configurations). The communication happens in sessions and, during a session, the client sends a FTP request for a file to the server and waits for the reply to complete before sending a new request. The connection gets closed if either simulation time limit is reached or if number of requests get complete. In our case, connection gets closed due to simulation time limit constraint, which is set to 500 seconds. Simulation parameters are chosen as those used earlier for the analytical model. Thus, the nodes are connected with the link capacity of 1000 $^{\text{packets}}/_{\text{sec}}$ (packet size of 1452 bytes) and propagation delay of 0.02 seconds. The downlink is represented with the path from server to client that is heavily loaded with the long-lived TCP flow, whereas the uplink path is lightly loaded as it contains only ACKs. All the nodes utilize RED queue mechanism with

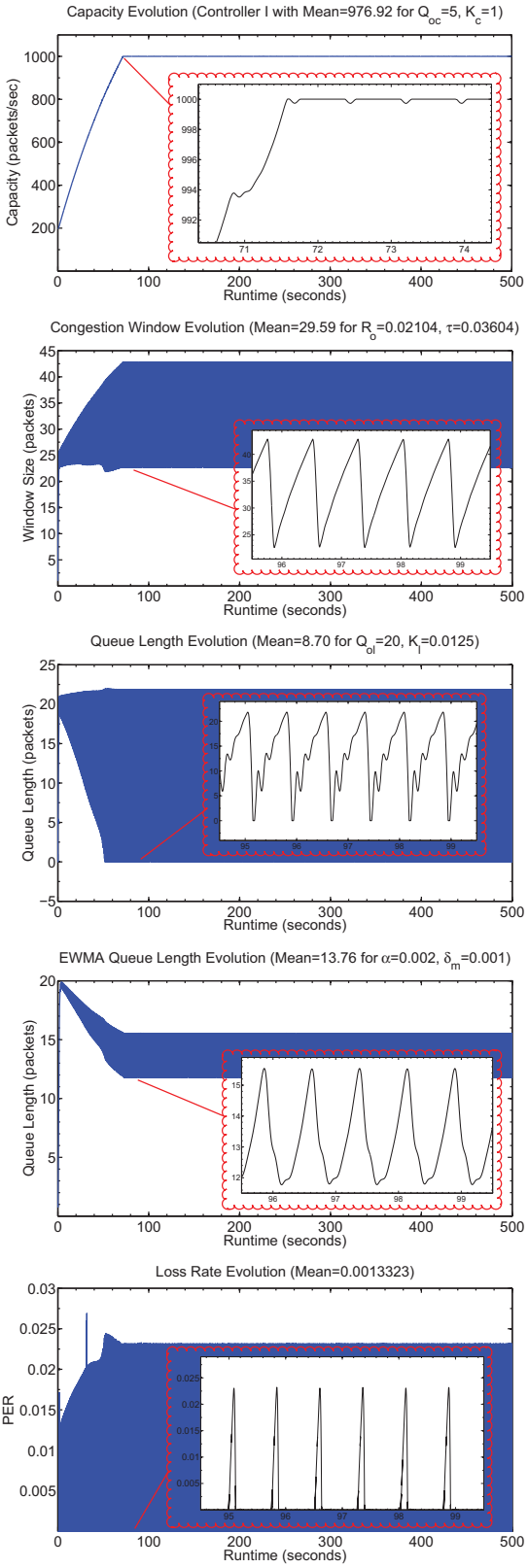


Fig. 1. Example of evolution of the parameters based on the fluid model.

lower and upper bound limits of 20 and 100 packets and linear growth of the packet discard probability from 0 to 1. The router along the path implements the rate adaptation algorithm at queues as detailed below.

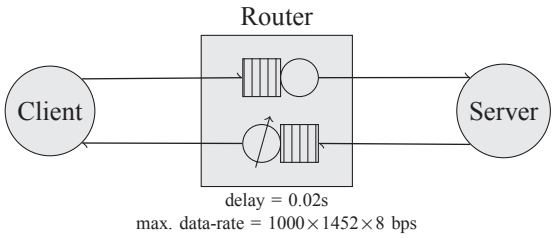


Fig. 2. The simulation topology.

*2) Rate Controller I:* To validate the fluid model based rate controller I presented in Section II-B1, we have implemented the equivalent rate control mechanism on the router using the OMNeT++ simulator. The implementation of rate control mechanism on link layer is mathematically described as the following,

$$C(t) = C(t) + K_c^{'}(Q(t) - Q_{oc})\delta_s \tag{8}$$

The parameters are chosen similar to mathematical model described earlier. Furthermore, the capacity is bounded in the region of $C_{min} = 100$ to $C_{max} = 1000$ $^{\text{packets}}/_{\text{sec}}$ with the initial starting capacity set to $C_0 = 200$ $^{\text{packets}}/_{\text{sec}}$. The $\delta_s$ in the simulator is the time interval wherein the controller checks the queue condition and updates the rate. To get optimal results, the value of $\delta_s$ should be less than the value of RTT. We have chosen $\delta_s$ to be small enough to react adequately towards the changes in queue length. This action can be compensated if the value of $K_c^{'}$ is chosen to be high enough. Indeed, the product of $K_c^{'}\delta_s$ is equivalent to the $K_c$ described in the fluid model.

Since the differential equation solution neglects a number of TCP mechanisms (slow start phase, timeouts, and so on), while the simulator has a detailed description of the TCP protocol, we expect to see some different behavior, especially during the initial transient phase due to slow start. Also the fact that the differential equation model provides solution as continuous increments, while the simulator works in discrete steps, is going to make the results from the two approaches different.

Figure 3 illustrates the behavior of rate controller evolution implemented using OMNeT++ simulator for different values of $K_c^{'}\delta_s$ along with its comparison with fluid model based rate controller. The comparison of results shows that the fluid model is very accurate, despite the different approaches as discussed above, and follows the trail of capacity evolution obtained through simulations.

*3) Rate Controller II:* We repeat the experiments described in previous section with the same parameters but for the rate controller, which now evolves based on EWMA queue length. The evolution of estimated queue length is computed using the following formula,

$$\bar{Q}(t) = (1 - \alpha)\bar{Q}(t) + \alpha Q(t) \tag{9}$$

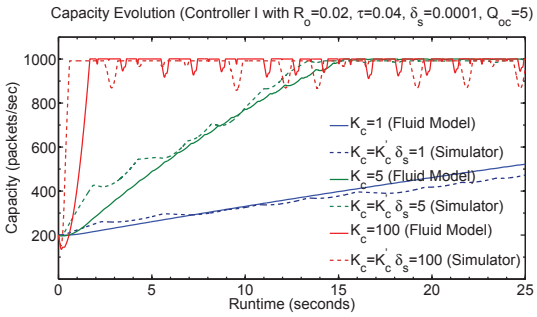


Fig. 3. Capacity controller based on instantaneous queue length.

The parameter $\alpha$ represents the smoothing factor ranging between 0 and 1, where the value $\alpha = 1$ represents the case of no averaging, i.e., of instantaneous queue length. Hence, (8) becomes,

$$C(t) = C(t) + K_c^{'}(\bar{Q}(t) - Q_{oc})\delta_s \tag{10}$$

Figure 4 shows the behavior of rate controller based on EWMA queue length and its comparison with fluid model based rate controller. The simulation results show that the model does quite well in capturing the dynamics of capacity evolution for different values of $K_c$.

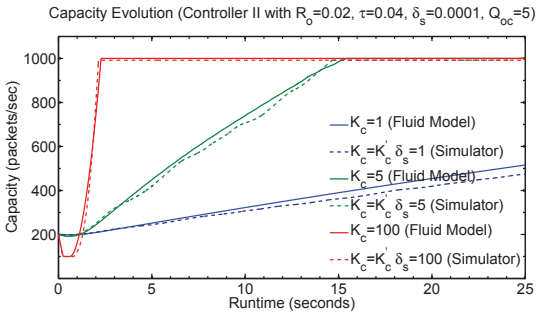


Fig. 4. Capacity controller based on average estimated queue length.

## IV. Interaction between TCP and Rate Adaptation

### A. Steady State Analysis

The steady state analysis of the fluid models allows us to analyze the fundamental relationships between the parameters of the model. To analyze the steady state of the designed rate adaptation scheme described in Section II-B1, we derive the equilibrium condition of the set of differential equations, (3), (4), (5) and observe its equilibrium state. At the steady state assuming $Q_{oc} < Q_{ol}$, when the derivatives are null, we have,

$$W = \sqrt{\frac{2}{L}}, \quad C = \frac{W}{RTT}, \quad Q = Q_{oc} \tag{11}$$

From these expressions, we observe that, as expected and as typical of TCP behavior, the window size depends on the inverse of the square root of the loss rate and adapts to the capacity through the RTT. Interestingly, the mean queue length settles to the value of $Q_{oc}$, that is the control threshold of capacity.

Considering that the capacity is bounded within $C_{min}$ and $C_{max}$ parameter, we have the following steady state equation for capacity, which basically shows two regimes:

$$C = \begin{cases} min\left(\frac{W}{RTT}, C_{max}\right), & Q_{oc} < Q_{ol} \\ max\left(\frac{W}{RTT}, C_{min}\right), & Q_{oc} > Q_{ol} \end{cases} \tag{12}$$

These expressions suggest that if the control of capacity kicks in before the control of TCP, that occurs through losses ($Q_{oc} < Q_{ol}$), the capacity stabilizes to the maximum one $C_{max}$, given the steady state capacity ($C = \frac{W}{RTT} > C_{max}$). However, given ($C_{min} < C = \frac{W}{RTT} < C_{max}$), the steady state capacity settles within the range of $C_{min}$ and $C_{max}$, i.e., capacity is scaled according to instantaneous offered load. If instead, losses occur when capacity controller has not yet entered into play ($Q_{oc} > Q_{ol}$), the TCP control loop shrinks the window and the TCP performance, without actually letting the capacity to grow. This eventually means the ratio $\frac{W}{RTT}$ becomes smaller and therefore, the capacity regime is forced to stabilize at the minimum bound $C_{min}$, given the steady state ($C = \frac{W}{RTT} < C_{min}$). A fundamental guideline to the design of the interaction between the rate adaptation and the TCP control loops is thus to make TCP control loop act when capacity control is already effective. In terms of parameter settings, this means making $Q_{oc}$ be smaller than $Q_{ol}$.

Extending the steady state analysis based to rate controller II is simple, and the conclusions are similar to those discussed above. The only difference is that instead of queue length ($Q$) settling around the $Q_{oc}$, the rate controller adjusts its capacity in such a way that the estimated queue length ($\bar{Q}$) settles around the value of $Q_{oc}$.

### B. Parameter Sensitivity

We report in this section the impact of parameters on the capacity evolution considering the case where the controller is based on instantaneous queue length, namely rate controller I. Sensitivity analysis based on rate controller II is omitted, as the conclusions drawn based on rate controller II are similar to the ones obtained from rate controller I. Note that the values of parameter settings for each simulation experiment are indicated in the title of the corresponding figure.

*Impact of $Q_{oc}$:* The parameter $Q_{oc}$ is the threshold for the rate adaptation capacity controller to scale the capacity. The value of $Q_{oc}$ is chosen to be less than $Q_{ol}$, so that before losses occur, capacity is allowed to increase and possibly avoid losses; a choice of $Q_{oc}$ larger than $Q_{ol}$ would make losses act on the congestion control of TCP before capacity is increased and would, thus, end up forcing the capacity to its minimum $C_{min}$ as seen in Section IV-A. The steady state analysis suggests that the capacity controller tries to adjust capacity such that the mean queue length is kept around the value of $Q_{oc}$. The smaller the value of $Q_{oc}$ is, the more aggressive

the capacity controller is in pushing capacity to its maximum $C_{max}$.

This is confirmed by the results of the experiments that are reported in Figure 5, representing the impact of $Q_{oc}$ on capacity evolution. Clearly, small values of $Q_{oc}$ lead to a more reactive control that make the capacity grow to the maximum in a short time. Observe also the case previously mentioned with $Q_{oc} > Q_{ol}$, that converges the capacity to its minimum.

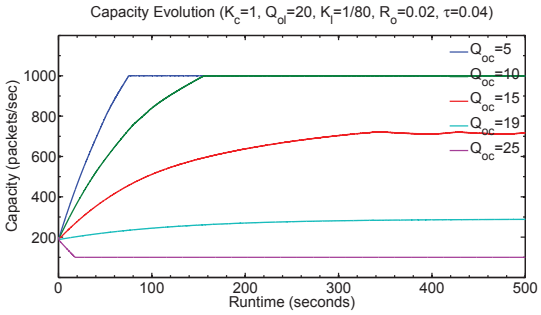


Fig. 5. Influence of $Q_{oc}$.

*Impact of $K_c$:* The capacity scaling parameter $K_c$ symbolizes how aggressive the capacity controller is towards the changes in queue length. Figure 6 illustrates the impact of $K_c$ on the evolution of the capacity, i.e., on the performance of the controller. For low values of $K_c$, the rate controller is slow in reacting and reaches maximum capacity in long time (order of 100 seconds). Large values of $K_c$ are therefore needed to avoid that the TCP performance deteriorates. Moreover for high values of $K_c$, the controller reacts much faster but it risks to follow too closely the variations of the queue length, as indicated by the large oscillations that can be seen in the zoomed small picture in the same figure.

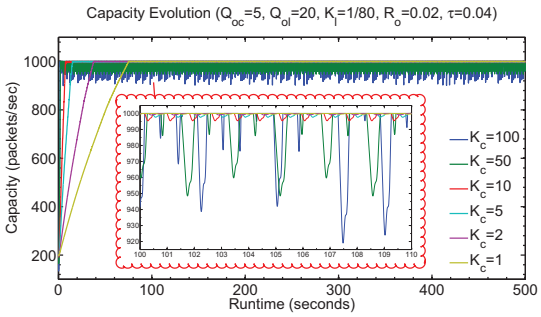


Fig. 6. Influence of $K_c$.

*Impact of $\tau$:* The parameter $\tau$ represents the amount of time need by the source to react to a loss indication. Usually, the time to react to losses in TCP is about the RTT, since losses are detected through the flow of ACKs. However, here, we vary $\tau$ to check how the time delay between the occurrence of losses and TCP reaction interact with the rate adaptation mechanism. We, thus, perform some experiments in which we let $\tau$ vary and collect the results in Fig. 7, that illustrates the impact of $\tau$ on the evolution of capacity. When $\tau$ is large, it means that TCP reacts with a large delay to some losses and then it becomes less aggressive in growing the congestion window. This implies that many losses occur in the meanwhile and TCP ends up reducing significantly the window size. This, in turn, means small queue length and small capacity. Notice indeed that for large $\tau$ (i.e., for large RTT), the system is not able to reach $C_{max}$.

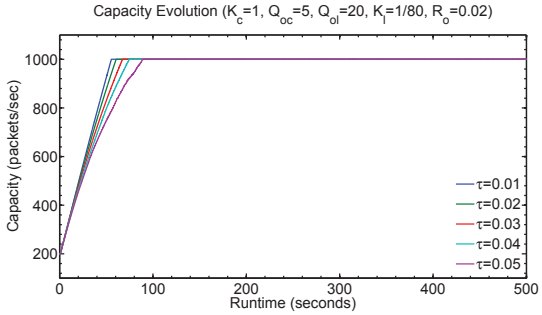


Fig. 7. Influence of $\tau$.

*Impact of $Q_{ol}$:* The parameter $Q_{ol}$ represents the value of the queue size at which losses start to occur. In terms of control, since losses act as input to the congestion control loop of TCP, $Q_{ol}$ represents the value of the queue at which the congestion control kicks in.

The impact of the parameter $Q_{ol}$ is shown in Fig. 8, that reports the capacity evolution for different values of $Q_{ol}$. High values of $Q_{ol}$ results in high values of the TCP window size, which eventually result in long queues. As previously discussed, when the value of $Q_{ol}$ is large the capacity controller can act on the capacity and reach the maximum before losses occur, i.e., before TCP control loop enters into play. The capacity controller becomes more aggressive for higher value of $Q_{ol}$, as the difference between $Q(t)$ and $Q_{oc}$ in (5) gets higher. This means that large values of $Q_{ol}$ means a more reactive controller, as shown in the figure. Overall the results are similar to the choice of $Q_{oc}$ seen in Fig. 5.

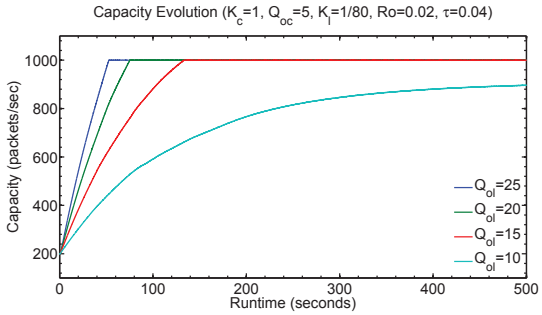


Fig. 8. Influence of $Q_{ol}$.

*Impact of $K_l$:* Analogous to $K_c$, the parameter $K_l$ is the loss scaling factor that represents the reactivity of the loss control. Higher values of $K_l$ result in higher losses, and lower TCP window size. Small window size, in its turn, implies small queue length and small capacity. For the sake of brevity, the figure is omitted here.

*Impact of $R_o$:* The propagation delay $R_o$ makes the control loop of TCP be more or less closely coupled with the queue with its own control loop in the rate adaptation. Large values of $R_o$ make the TCP congestion control react slowly to the queue state and this ends up inducing oscillations to the queue length and transmission rate. Results are similar to the ones observed for the variation of $\tau$.

### C. Average Capacity

To further investigate the interplay between rate adaptation and TCP control loop, we focus on the average values of capacity taken from each simulation run of 200 seconds duration. The measure is accounting the initial transient growth as well as the steady state of capacity. We consider the rate controller I and let the parameter $Q_{oc}$ vary between 5 to 25 while keeping the losses constant ($Q_{ol} = 20$ and $K_l = 1/80$). Again, the rate controller evolves based on the same values of $C_{min} = 100$, $C_{max} = 1000$, and $C_0 = 200$. Results are reported in Fig. 9 for different values of $K_c$; to emphasize the relationship between $Q_{ol}$ and $Q_{oc}$, we report on the x-axis the values $Q_{ol} - Q_{oc}$.

We observe that the capacity reaches $C_{min}$ for $Q_{oc} > Q_{ol}$, whereas the region with $Q_{ol} - Q_{oc} > 0$ represents to those cases in which the capacity controller reacts to the changes in queue length before the congestion controller kicks in. It is interesting to observe that the value $Q_{ol} - Q_{oc}$ seems to have a stronger impact on results than the value of $K_c$, suggesting that the understanding of interplay between TCP and rate adaptation control loops is extremely important; carefully controlling the interaction between the two controls allows to achieve larger capacity, while acting on the rate adaptation alone might be useless.

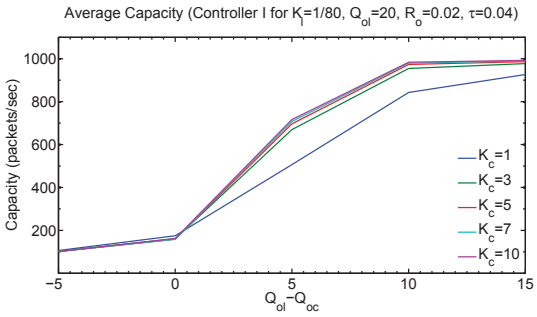


Fig. 9. Behavior of average transient plus steady state capacity.

## V. Comparison between the controllers

We make in this section a direct comparison between the performance of the two proposed rate adaptation controllers. To do so, we run the analytical models for 500 seconds and we take the average values of various parameters, such as the capacity, queue length and loss rate; the computed averages include the transient phase as well as the steady state phase. Figure 10 shows the comparison of the results for the two controllers, for a wide range of values of $K_c$.

The results show that the rate controller based on estimated queue length yields to smoother and more stable results with respect to the rate controller based on instantaneous queue length. While this is quite intuitive, it is important to emphasize that this smoother behavior is achieved at the cost of larger average queue length and larger amount of losses, that might, in general, in their turn deteriorate the QoS (higher delay variance, possibility of multiple losses and therefore difficulties in reacting to losses, and so forth). However, the differences are not that large in absolute terms.

Finally, for very large values of $K_c$, the controller I suffers from much larger oscillations than controller II (see also Fig. 6). This results in decrease of average capacity.

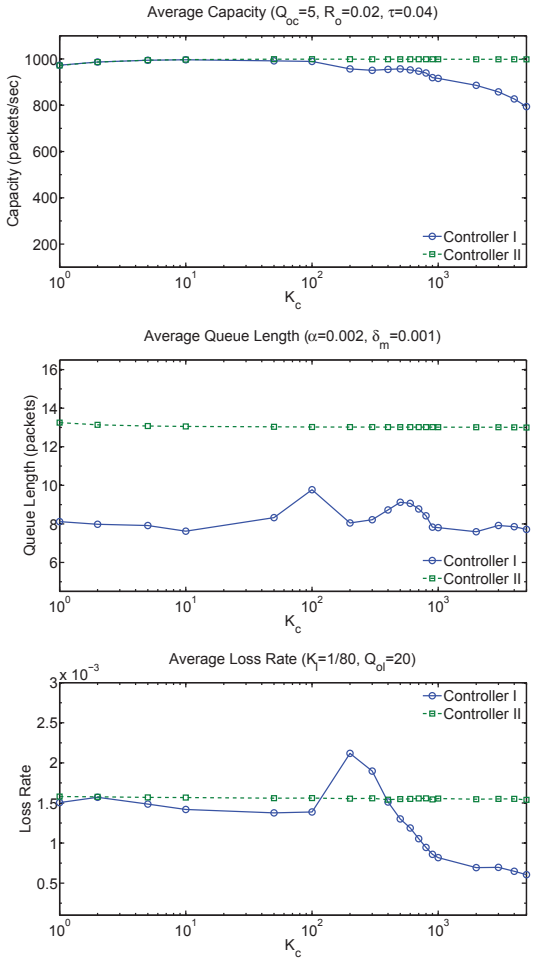


Fig. 10. Comparison of average capacity, queue length, loss rate.

## VI. Conclusion

In this paper, we have proposed a mathematical model to analyze the interaction between TCP congestion control and rate adaptation schemes implemented at the routers to save energy. The models are based on differential equations and simulation experiments have validated the models as being accurate and suitable to predict the system performance. The model accurately approximates TCP behavior despite the fact

that a number of TCP mechanisms, like timeouts and slow start, were disregarded.

Results indicate that the interaction between the rate controller and TCP is quite critical, and improper setting of the parameters might lead to bad performance. It might indeed happen that low rates are used in the router, with corresponding low throughput for the connection. Fine tuning of the rate controller parameter $K_c$ is needed. When properly tuned, the controller can be stable, leading to high capacity when needed with good resulting performance for TCP, confirming that rate adaptation can reduce energy consumption while maintaining high QoS, but at the cost of careful mechanism design.

## ACKNOWLEDGMENTS

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 257740 (Network of Excellence TREND).

## REFERENCES

[1] R. Bolla, R. Bruschi, F. Davoli, and F. Cucchietti, "Energy Efficiency in the Future Internet: A Survey of Existing Approaches and Trends in Energy-Aware Fixed Network Infrastructures," *Communications Surveys Tutorials, IEEE*, vol. 13, no. 2, pp. 223–244, 2011. [Online]. Available: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5522467

[2] P. Reviriego, A. Sanchez-Macian, and J. A. Maestro, "On the Impact of the TCP Acknowledgement Frequency on Energy Efficient Ethernet Performance," in *Proceedings of the IFIP TC 6th international conference on Networking*, ser. NETWORKING'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 265–272. [Online]. Available: http://dl.acm.org/citation.cfm?id=2039912.2039941

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

[4] S. Nedevschi, L. Popa, G. Iannaccone, S. Ratnasamy, and D. Wetherall, "Reducing Network Energy Consumption via Sleeping and Rate-Adaptation," in *Proceedings of the 5th USENIX Symposium on Networked Systems Design and Implementation*, ser. NSDI'08. Berkeley, CA, USA: USENIX Association, 2008, pp. 323–336. [Online]. Available: http://dl.acm.org/citation.cfm?id=1387589.1387612

[5] C. Gunaratne, K. Christensen, B. Nordman, and S. Suen, "Reducing the Energy Consumption of Ethernet with Adaptive Link Rate (ALR)," *Computers, IEEE Transactions on*, vol. 57, no. 4, pp. 448–461, April 2008.

[6] M. Allman, V. Paxson, and E. Blanton, "TCP Congestion Control," IETF RFC 5681, Tech. Rep., Sept. 2009. [Online]. Available: http://tools.ietf.org/html/rfc5681

[7] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based Analysis of a Network of AQM Routers Supporting TCP Flows with an Application to RED," in *IN PROCEEDINGS OF ACM SIGCOMM*, 2000, pp. 151–160.

[8] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 397–413, Aug. 1993. [Online]. Available: http://dx.doi.org/10.1109/90.251892