A Mathematical Model of the Skype VoIP Congestion Control Algorithm

Luca De Cicco and Saverio Mascolo

***Abstract*—Voice over Internet Protocol (VoIP) is an Internet application of ever increasing importance. The purpose of this note is to derive a mathematical model of the Skype VoIP congestion control. The proposed model is in the form of a non linear hybrid automaton, which has been validated through extensive experiments. The dynamics of the Skype sending rate, the stability of its equilibrium points and the efficiency in bandwidth utilization while avoiding network instability are analyzed. Results show that, under network congestion, the Skype sending rate is driven by the packet loss ratio and matches the available bandwidth with a steady state finite error.**

***Index Terms*—Hybrid automaton, Computer networks, VoIP congestion control**

I. Introduction

The transport of multimedia traffic over the Internet is of ever increasing importance due to the spreading of multimedia content delivery applications such as IP television (IPTV), Videoconferencing over IP, Voice over IP (VoIP), video on demand (VoD).

The stability of the traditional data Internet is due to the Transmission Control Protocol (TCP) congestion control algorithm, which constitutes the most complex part of the TCP [1]. However, the TCP congestion control is not appropriate to deliver time-sensitive traffic such as VoIP calls, because of its retransmission mechanism and additive increase multiplicative decrease (AIMD) sliding window control [2], [3]. As a consequence, audio/video applications employ proprietary congestion control algorithms that are executed over the UDP protocol, which is a transport protocol that does not implement congestion control [2].

Voice over IP is an important example of a real-time service increasingly offered over the Internet. Skype VoIP today counts over 40 millions active users worldwide and up to 17 millions concurrent users[1]. Skype reports that more than 100 billions minutes of calls have been generated by Skype clients, resulting in the most used VoIP application[2]. Thus, Skype can be considered as the most representative application producing VoIP traffic. This explosive growth poses challenges to telecom operators and ISPs both from the point of view of network stability and business model. For this reason, it is important to assess the impact of a large number of Skype VoIP calls on TCP responsive traffic that stills delivers the most part of the Internet traffic [2], [3]. In second instance, this study is useful to understand if there is the need for a better designed congestion control algorithm for VoIP.

Up to now, the only congestion control algorithm for data networks that has been widely modelled and analyzed is the standard TCP congestion control and its variants [4],[5],[6],[7],[8]. This is due to the fact that the TCP congestion control algorithm and its variants are fully disclosed and well described in scientific literature and in standardization bodies such as the IETF.

The TCP Friendly Rate Control (TFRC) protocol is a new congestion control algorithm designed for multimedia flows which is under standardization within the IETF [3]. However, as a matter of fact, this standardization effort is lagging behind and TFRC is not used in any commercial multimedia application.

Luca De Cicco is post-doc at Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari, Via Orabona 4, Italy (e-mail: ldecicco@gmail.com)

Saverio Mascolo is a Faculty member of Dipartimento di Elettrotecnica ed Elettronica, Politecnico di Bari, Via Orabona 4, Italy (e-mail: mascolo@poliba.it)

[1]http://share.skype.com/stats_rss.xml

[2]http://ebayinkblog.com/wp-content/uploads/2009/01/skype-fast-facts-q4-08.pdf

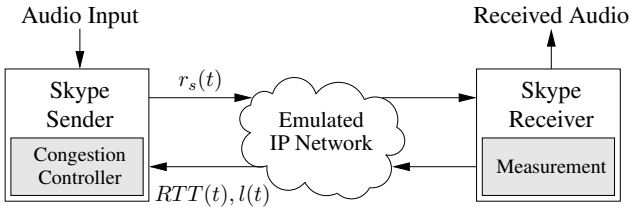

Fig. 1. Experimental testbed emulating a Skype audio call over the Internet

The literature regarding Skype can be grouped in three main categories: 1) studies concerning its peer-to-peer overlay network [9], [10]; 2) studies on its traffic characteristics (see [11], [12] and references therein); 3) studies on how to identify Skype calls among other flows [13].

Authors of [9] present the results of a large-scale experimental evaluation focusing on the P2P network employed by Skype. In particular they focus on the technique employed to implement NAT traversal functionalities and they characterize the super-nodes (SN) of the P2P network.

In [11], [12], we have shown that Skype reacts to network congestion by reducing the sending rate, thus being able to match the link capacity to some extent.

In [13] the characteristics of Skype VoIP traffic have been studied and a method to identify Skype calls among other flows has been proposed. The method is based on two classifiers: one employs the Chi-Square test to the payload of passively sniffed traffic, the other measures packet size and inter packet gaps.

This paper enriches the literature by proposing for the first time a mathematical model that describes how Skype VoIP dynamically tracks the network available bandwidth in the presence of variable network conditions. The model can be used to analyze fundamental properties of the Skype congestion control such as network stability and efficiency in network utilization.

Modelling the sending rate produced by Skype is difficult due to the following reasons: 1) the protocol behaviour is hidden by AES encryption; 2) the input variables that drive the controller are unknown; 3) it is very reasonable to conjecture that the controller implements switching dynamics due to the use of if-then-else decisional blocks.

The rest of this note is organized as follows: in Section II we summarize the data collected through extensive experimental investigations in order to build the model; in Section III we propose a mathematical model of Skype sending rate when congestion occurs; in Section IV we derive a hybrid model of a Skype flow accessing a bottleneck link along with a stability analysis. Finally, Section V draws the conclusions.

II. Experimental Testbed and Background

Fig. 1 shows a block diagram of the system to be modelled. We consider the Skype sending rate dynamics as the output of a black box that models the congestion controller. In [14] we have conjectured the following two input variables: 1) the connection round trip time (RTT); 2) the packet loss ratio. To the purpose, it is worth noting that round trip times and packet loss rates are two variables that can be easily measured end-to-end to be used as feedback signals in end-to-end congestion control algorithms. In particular, congestion control algorithms that employ delay measurements to adjust the input rate are called *delay-based* (such as in the case of TCP Vegas and TCP Fast), whereas algorithms that react to losses are called *loss-based* (such as in the case of TCP Reno and its variants).

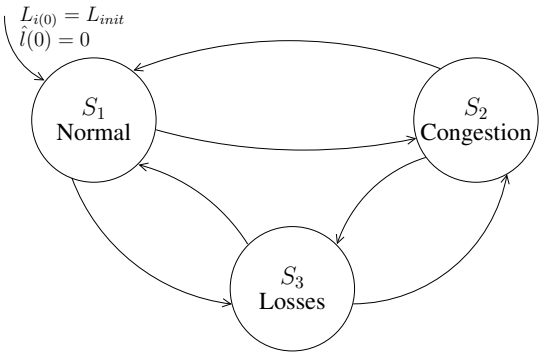

Fig. 2. Skype VoIP hybrid system model

The Skype VoIP sender receives round trip time $RTT(t)$ and loss ratio $l(t)$ as feedback variables[3] and set the sending rate $r_s(t)$ by throttling both packet size and packet sending rate.

In order to model how feedback signals drive the Skype sending rate, we have collected data through extensive experimental measurements obtained using the testbed shown in Fig. 1, which consists of real Skype applications running over a local area network where a router-host is properly equipped to delay packets, set the available bandwidth $b(t)$ and set packet drop probability $l(t)$. In particular, we have considered step-like inputs for $RTT(t)$, $l(t)$ and the available bandwidth $b(t)$, which is a basic practice when trying to investigate the dynamic behaviour of a system. The experimental evaluation reported in [14] has led to the following findings: 1) variations of the input signal $RTT(t)$ have no significant effect on the input rate, thus Skype VoIP does not implement a delay-based congestion control; 2) when the signal $l(t)$ is increased, Skype reacts to a persistent loss by increasing the transmission rate; moreover, in such cases, packet size is increased, suggesting that Skype employs a Forward Error Correction (FEC) scheme in order to be able to recover lost packets. FEC is a widely employed technique aimed at recovering lost packets in real-time Internet applications such as VoIP. Basically, the idea is to piggyback redundant information, i.e. the FEC data, on a VoIP packet in order to be able to recover lost packets. The redundant information is obtained by computing the XOR of previously sent VoIP packets [15].

In [14] we have shown that the Skype VoIP sending rate can be modelled using the *hybrid automaton* shown in Fig. 2.

In particular, the sending rate can exhibit three different dynamics depending on the state S_i ($i = 1, 2, 3$) of the automaton: in the state S_1, which is characterized by normal network conditions, i.e. no congestion occurs and no losses are present, the sending rate is kept constant; in the state S_2, which is triggered when Skype realizes that the network available bandwidth has changed and congestion occurs, the sending rate is throttled by using the congestion control algorithm; in the state S_3, which is triggered when losses not due to network congestion are present (i.e. due to a lossy link), Skype employs a Forward Error Correction (FEC) scheme in order to counteract the losses attributed to a lossy link. In [14] we have found that the loss ratio measured by Skype $\hat{l}(t)$ is a first-order low passed version of the actual loss ratio $l(t)$ experienced by the flow. The time constant of the filter was found to be around 11 s.

In the next Section, we will focus on the behaviour of the Skype VoIP sending rate in the presence of variations of the available bandwidth.

[3]Skype shows measured RTT and loss ratio in its "technical info" tool-tip

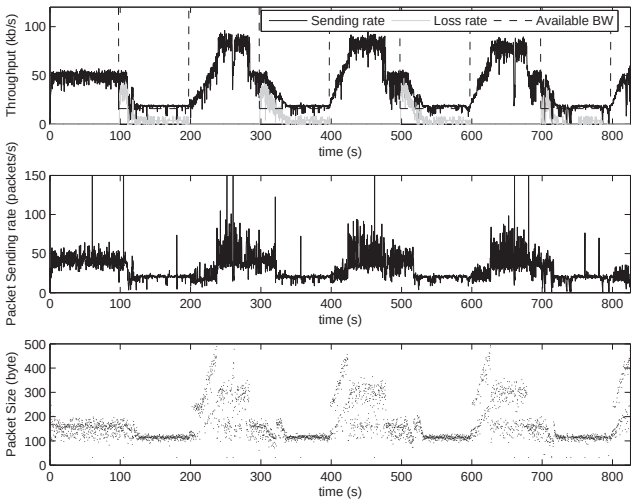

Fig. 3. Sending rate, loss rate, packet sending rate and packet size evolution in the presence of a square wave available bandwidth of 200 s period

III. The Skype Congestion Control Model

This Section aims at providing a mathematical model describing the dynamic behaviour of the Skype sending rate in the presence of a variable link available bandwidth.

To the purpose, we investigate how the Skype sending rate reacts to sudden changes of available bandwidth. We employ an available bandwidth that varies as a square wave with maximum value $A_M = 160$ kb/s, minimum value $A_m = 16$ kb/s and with a period equal to 200 s, which is large enough to show all the transient dynamics.

Fig. 3 shows that Skype decreases the sending rate when the link capacity drops from the value A_M to the value A_m. The Skype flow takes approximately 40 s to track the available bandwidth during which it experiences a significant loss rate. It can be viewed that, when the available bandwidth drops, the loss rate increases to a peak value of 35 kb/s whereas the sending rate reduces to less than 20 kb/s in around 40 s. When the link capacity increases (i.e. at $t = 400$ s) the input rate ramps up to 90 kb/s, again in around 40 s.

In order to find out the Skype controller dynamics, we have captured both packet sizes and packet sending rate over time (see Fig. 3). It can be noticed that the packet sending rate drops in a step-like function when Skype detects congestion, whereas the packet size is decreased slowly, resulting in a slow adaptation to the available bandwidth. In other terms, the most part of Skype congestion control is performed by throttling the packet sending rate, whereas the packet size is changed for fine tuning.

Based on the large number of experiments we have run [11], [14], we make the hypothesis that the audio codec employed by Skype is multi-rate so that the encoder can select among N levels $L_k = \{L_1, L_2, \ldots, L_N\}$ with $L_1 < L_2 < \cdots < L_N$. Moreover, we assume that the Skype adaptive codec is able to select the most appropriate mode according to a metric which is the analogous of Carrier to Interference ratio (C/I) in the case of Adaptive Multi-Rate Wide Band (AMR-WB) encoder.

By letting $i(t)$ denote the switching law $i(\cdot) : \mathbb{R} \mapsto \{1, \ldots, N\}$ and $L_{i(t)}$ the encoder level at time t, we are ready to make the key hypothesis that the Skype control law for the sending rate is ruled by the following equation:

$$r_s(t) = (1 - \hat{l}(t)) \cdot (1 + f(t))L_{i(t)} \quad (1)$$

where $f : \mathbb{R} \mapsto [0, 1] \subset \mathbb{R}$ models the FEC action, meaning that when $f(t) = 0$ the FEC action is not active whereas when $f(t) = 1$

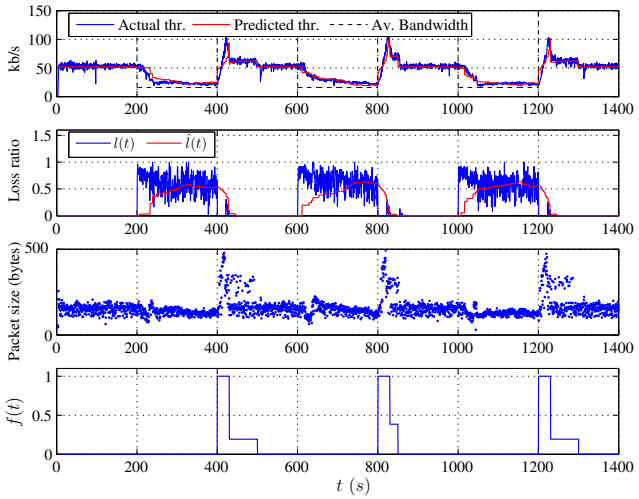

Fig. 4. Comparison between the actual and predicted rate using (1)

the FEC action is at maximum. Equation (1) describes the following behaviours: when $\hat{l}(t) = 0$ (i.e. no congestion occurs) the sending rate $r_s(t)$ is given by $L_{i(t)}$ or $(1 + f(t))L_{i(t)}$ if the FEC is active. When congestion happens (i.e. $\hat{l}(t) \neq 0$) the feedback signal reduces the sending rate of $\hat{l}(t)(1 + f(t))L_{i(t)}$.

We have run a set of experiments using a square wave available bandwidth with maximum value of 160 kb/s, a minimum value of 16 kb/s with a period of 400 s.

Fig. 4 shows that the packet size is kept roughly constant until the first bandwidth increase occurring at $t = 400$ s. We conjecture that Skype infers an increase of the available bandwidth by measuring a decrease in both RTT and $l(t)$, which triggers a "probing phase". Since losses can occur during this phase, Skype activates the FEC action. We conjecture that Skype implements the FEC action by bundling multiple frames in a single packet, so that an increase in the FEC action will turn into a larger packet size [12], [15]. For these considerations, we conjecture the FEC action $f(t)$ shown in Fig. 4, which is proportional to the packet size. Fig. 4 shows that the FEC action is reduced when the filtered loss ratio $\hat{l}(t)$ reaches zero, and it is turned off later when $\hat{l}(t)$ is detected not to grow. Moreover, by looking at the packet size evolution, we infer that the FEC action is kept off during congestion.

Fig. 4 shows both the Skype measured and predicted sending rate using (1) with a constant value of $L_{i(t)}$ equal to 56 kb/s. This value is measured at the beginning of the Skype connection when $f(t) = 0$ and $\hat{l}(t) = 0$.

Equation (1) nicely predicts the Skype sending rate, both in the case of presence or absence of congestion, including the large transient time we have reported before.

In conclusion, the main driver of the Skype sending rate is $\hat{l}(t)$, which is a low-pass filtered version of the actual loss ratio $l(t)$. Therefore, the dynamics of $\hat{l}(t)$ affects the sending rate response to congestion which is somewhat slow.

IV. Hybrid Automaton Modelling a Skype Flow Accessing a Bottleneck

In this Section we propose a hybrid automaton to model a Skype flow accessing a single bottleneck link characterized by an available bandwidth $b(t)$, a drop tail queue whose maximum size is q_M and a round trip time T which is the sum of the delay of the forward path T_1 and the delay of the backward path T_2 .

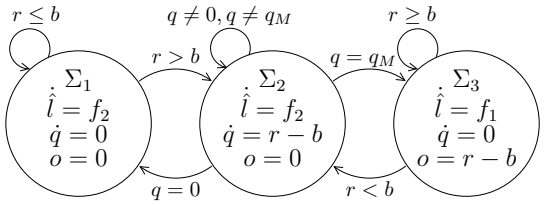

Fig. 5. The hybrid automaton model $\mathcal{H}$ of a Skype flow accessing a drop tail queue

The queue dynamics can be modelled by the following differential equation [16]:

$$\dot{q}(t) = \begin{cases} 0 & q(t) = 0, r(t) \leq b(t) \text{ or} \\ & q(t) = q_M, r(t) \geq b(t) \\ r(t) - b(t) & \text{otherwise} \end{cases} \quad (2)$$

where $r(t)$ is the queue input rate. The queue overflow rate $o(t)$ can be modelled as follows:

$$o(t) = \begin{cases} r(t) - b(t) & q(t) = q_M, r(t) > b(t) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

which means that when the queue is full the exceeding input rate $r(t) - b(t)$ is dropped [16].

The dynamic model of the Skype sending rate can be derived from (1) by considering that the feedback signal $\hat{l}(t)$ is a filtered version of the actual packet loss ratio $l(t) = o(t)/r(t)$ measured at the sender after the backward delay T_2. Based upon experiments, we assume that Skype filters $l(t - T_2)$ using a first order low pass filter with a time constant τ:

$$\dot{\hat{l}}(t) = -\frac{1}{\tau}\hat{l}(t) + \frac{1}{\tau}l(t - T_2)$$

which, by considering that $l(t) = o(t)/r(t)$, turns out:

$$\dot{\hat{l}}(t) = -\frac{1}{\tau}\hat{l}(t) + \frac{1}{\tau}\frac{o(t - T_2)}{r(t - T_2)} \quad (4)$$

By substituting (1) and (3) in (4), after straightforward computations we obtain:

$$\dot{\hat{l}}(t) = \begin{cases} f_1 = \frac{1-\hat{l}(t)}{\tau} - \frac{b(t-T_2)/\tau}{(1-\hat{l}(t-T_2))(1+f(t-T_2))L_{i(t-T_2)}} & q = q_M, \\ & r > b \\ f_2 = -\frac{1}{\tau}\hat{l}(t) & \text{otherwise} \end{cases} \quad (5)$$

It is simple to show that the dynamics of the considered system can be described by means of the three states hybrid automaton $\mathcal{H}$ which is shown in Fig. 5.

Let $X = \{\hat{l}, q : 0 \leq \hat{l} \leq 1, 0 \leq q \leq q_M\} \subset \mathbb{R}^2$ be the set of continuous states, $x(t) = \begin{bmatrix} \hat{l}(t) & q(t) \end{bmatrix}^T \in X$ be the continuous state of the system at the continuous time t and $\Sigma = \{\Sigma_1, \Sigma_2, \Sigma_3\}$ denote the discrete set of $\mathcal{H}$ [17]. In particular: the state Σ_1 holds when the queue is empty and the input rate is less then or equal to the link available bandwidth; the system dynamics is described by state Σ_2 when the queue is neither full nor empty; the state Σ_3 describes the evolution of the system when the queue is full and the input rate is greater than or equal to the available bandwidth [16]. By considering the guard conditions ($r > b$ and $r < b$) the domains (or invariant sets) of the three discrete states are the following:

$$\begin{aligned} Dom(\Sigma_1) &= \{x \in X | q = 0 \wedge 1 - \frac{b}{(1+f)L_i} \leq \hat{l} \leq 1\} \\ Dom(\Sigma_3) &= \{x \in X | q = q_M \wedge 0 \leq \hat{l} \leq 1 - \frac{b}{(1+f)L_i}\} \\ Dom(\Sigma_2) &= X \setminus (Dom(\Sigma_1) \cup Dom(\Sigma_3)) \end{aligned}$$

In the sequel we will prove four lemmas that will be used to analyze the stability of the equilibrium points of the automaton $\mathcal{H}$.

Before starting to prove the first Lemma, we notice that the filter time constant $\tau \cong 11\,\mathrm{s}$ is from 10 up to 100 times the value of the RTT measured in realistic Internet scenario, which goes from milliseconds up to few seconds such as in the case of old wireless GPRS networks or satellite connections. Moreover, ITU-T recommends a maximum value of 400 ms for mouth-to-ear delay [18], which corresponds to a round trip time of 800 ms, in order to consider the quality of a VoIP call acceptable. Therefore, since $\tau \gg T_2$ it is well justified to neglect the time delay T_2 in the model of the system.

Lemma 1: When the automaton is in the state Σ_3, the following inequality holds:

$$b(t) \leq L_{i(t)}(1 + f(t)) \tag{6}$$

Proof: In the state Σ_3, $b(t) \leq r(t)$ holds. Since $r(t) = (1 - \hat{l}(t))L_{i(t)}(1 + f(t)) \leq L_{i(t)}(1 + f(t))$, the inequality (6) is proved. Thus, if (6) is not verified $Dom(\Sigma_3)$ is an empty set, and Σ_3 is not reachable. ∎

Lemma 2: The system Σ_3 has a unique asymptotically stable equilibrium point that is feasible:

$$\begin{aligned} \hat{l}^* &= 1 - \sqrt{\tfrac{b^*}{L^*(1+f^*)}} \\ q^* &= q_M \end{aligned} \tag{7}$$

where b^*, L^*, f^* are the inputs at steady state.

Proof: The state Σ_3 holds when $r(t) \geq b(t)$ and the queue is full ($q = q_M$), i.e. when congestion occurs. Equation (5) in steady state condition (i.e. when $\dot{\hat{l}}(t) = 0$ and $\dot{q}(t) = 0$) gives the equilibrium point (7) where b^*, L^* and f^* are the steady state inputs. Since from Lemma 1 when in Σ_3 it holds $b^* \leq L^*(1 + f^*)$, it results that $\hat{l}^* \in [0, 1]$, i.e. the equilibrium is feasible.

In order to prove the stability of the equilibrium, we employ the direct Lyapunov method. By using the candidate Lyapunov function $V(\hat{l}) = \frac{1}{2}(\hat{l} - \hat{l}^*)^2$ the following derivative of $\dot{V}(\hat{l})$ is obtained after straightforward computations:

$$\dot{V}(\hat{l}) = -\frac{1}{\tau}\frac{(\hat{l} - \hat{l}^*)^2(2 - \hat{l} - \hat{l}^*)}{1 - \hat{l}}$$

which is definite negative for all $\hat{l} \in [0, 1]$. Thus, for the direct Lyapunov stability criterion the equilibrium $\hat{l}^*$ is asymptotically stable. ∎

Lemma 3: The system Σ_1 has a unique equilibrium point:

$$\begin{aligned} \hat{l}^* &= 0 \\ q^* &= 0 \end{aligned} \tag{8}$$

which is globally asymptotically stable.

Proof: The lemma is proved by observing that Σ_1 is a linear system with one real strictly negative eigenvalue. ∎

Lemma 4: The hybrid automaton $\mathcal{H}$ has a sink state Σ_3 if $b^* < (1 + f^*)L^*$ and a sink state Σ_1 if $b^* > (1 + f^*)L^*$.

Proof: Let us consider the first part of the proposition which assumes $b^* < (1 + f^*)L^*$. The proof starts by showing that for any initial condition $(\hat{l}_0, q_0) \in X$, the state dynamics of the hybrid automaton enters the state Σ_3 and remains indefinitely in this state, i.e. Σ_3 is a *sink* state. To this purpose we find the reachability set of $\mathcal{H}$.

Fig. 6 (a) shows that $Dom(\Sigma_2)$ is partitioned in the following two zones depending on the sign of $\dot{q}$:

$$\begin{aligned} Z_1 &= \{x \in X : \hat{l} < 1 - b^*/(1 + f^*)L^*, 0 \leq q < q_M\} \\ Z_2 &= \{x \in X : \hat{l} > 1 - b^*/(1 + f^*)L^*, 0 < q \leq q_M\} \end{aligned}$$

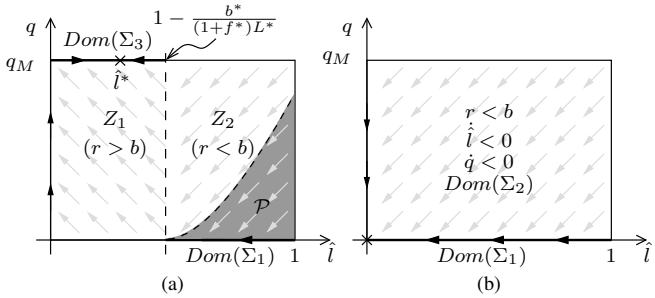

Fig. 6. Qualitative phase portrait of the Skype state space model when (a) $b^* < (1 + f^*)L^*$ or (b) $b^* \geq (1 + f^*)L^*$

Thus, it results that if $r > b$, $x \in Z_1$ then the queue builds up otherwise $x \in Z_2$ and the queue is drained.

It is easy to show that if we initialize $\mathcal{H}$ in the state Σ_1 (the queue is empty and $r < b$) then from (5) the state evolution is given by:

$$\begin{aligned} \hat{l}(t) &= \hat{l}_0 e^{-\frac{t}{\tau}} \\ q(t) &= 0 \end{aligned}$$

where the initial condition is $x_0 = (\hat{l}_0, 0) \in Dom(\Sigma_1)$. The state of $\mathcal{H}$ remains in Σ_1 if $r(t) < b(t)$ (i.e. when $\hat{l}(t) > 1 - b^*/(1 + f^*)L^*$). The state switches to Σ_2 at the time t_{12} when the sending rate matches the available bandwidth, i.e. when $r(t_{12}) = b^*$:

$$t_{12} = -\tau \log\left(\frac{1}{\hat{l}_0}(1 - \frac{b^*}{(1 + f^*)L^*})\right)$$

that is positive for each $x_0 \in Dom(\Sigma_1)$. Thus, we can conclude that if we initialize $\mathcal{H}$ in the Σ_1 state, the only reachable state is Σ_2 after a time t_{12}. Moreover, since the state is now in the region Z_1 the queue must build up, so that the only reachable state from Σ_2 is now Σ_3. Thus, we can conclude that if $\mathcal{H}$ is initialized in Σ_1 the only possible evolution of the state is $\Sigma_1 \to \Sigma_2 \to \Sigma_3$.

Let us now focus on the states that can be reached by initializing $\mathcal{H}$ in Σ_2, that is $(\hat{l}_0, q_0) \in Dom(\Sigma_2) = Z_1 \cup Z_2$. By solving (5), for $t > 0$ the evolution of the state is given by:

$$\hat{l}(t) = \hat{l}_0 e^{-\frac{t}{\tau}} \tag{9}$$

$$q(t) = q_0 - \hat{l}_0 \tau L^*(1 + f^*)(1 - e^{-\frac{t}{\tau}}) + (L^*(1 + f^*) - b^*)t \tag{10}$$

It is easy to show that if the initial condition belongs to Z_1, there is no way for the state to end in Σ_1 (since the queue must build up) so that the only state that can be reached is Σ_3 (full queue). Let us now start from the initial condition in Z_2. Now two different evolutions of the hybrid automaton are possible: the state can either go in Σ_1, then back to Σ_2 and then to Σ_3, or either go directly in Σ_3. Finally, if the initial condition belongs to the set:

$$\begin{aligned} \mathcal{P} = \{x \in Z_2 : q < \tau(L^*(1 + f^*) - b^*) \log\left(1 - \frac{b^*}{L^*(1 + f^*)}\right) \\ + \tau L^*(1 + f^*)\hat{l} - \tau(L^*(1 + f^*) - b^*)(1 + \log \hat{l})\} \subset Z_2 \end{aligned}$$

the state will follow the path $\Sigma_2 \to \Sigma_1 \to \Sigma_2 \to \Sigma_3$, otherwise it will follow the path $\Sigma_2 \to \Sigma_3$. In either the cases if the hybrid automaton starts from Σ_2 the state will end in Σ_3 that is, the queue fills up and packet losses occur.

In order to conclude the first part of the proof we need to show that if we initialize the system in Σ_3 the state of $\mathcal{H}$ will remain in Σ_3. This follows from Lemma 2 since the equilibrium (7) is asymptotically stable.

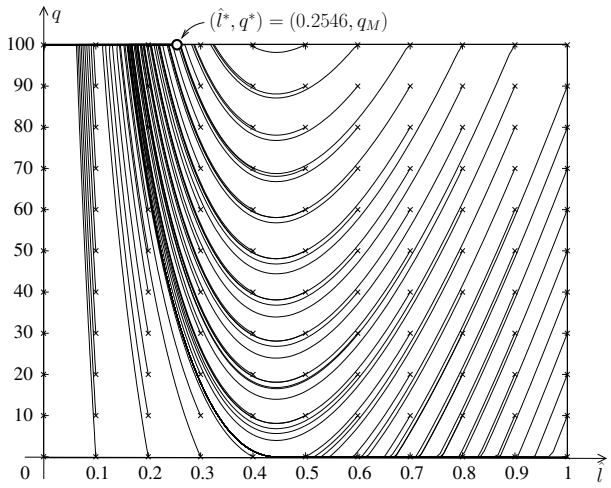

Fig. 7. Trajectories obtained using the proposed model when $L^* = 54\,\text{kb/s}$ and $b^* = 30\,\text{kb/s}$

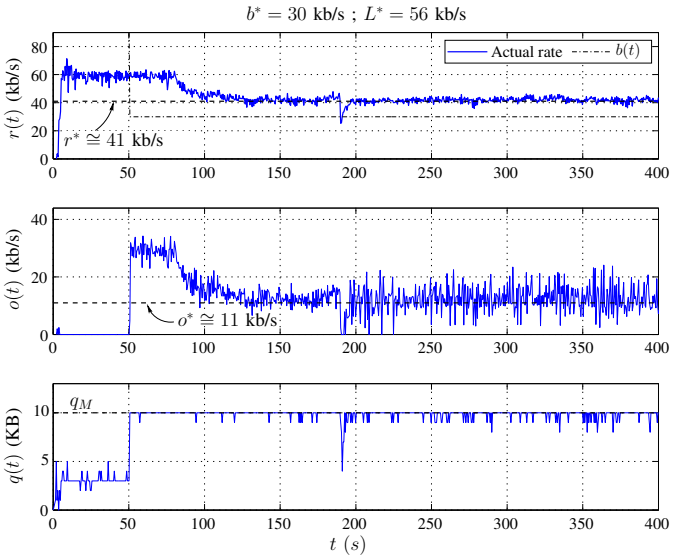

Fig. 8. Validation of the Skype VoIP model (Propositions 1 and 2)

The second part of the proof which states that Σ_1 is a sink state if $b^* > (1+f^*)L^*$ follows the same arguments we have developed in the first part and it is omitted due to space limitation. ■

Proposition 1: By considering the equilibrium inputs b^*, L^* and f^*, the hybrid automaton shown in Fig. 5 reaches the following asymptotically stable equilibrium state:

$$\begin{aligned} \hat{l}^* &= 1 - \sqrt{\tfrac{b^*}{L^*(1+f^*)}} \\ q^* &= q_M \end{aligned} \tag{11}$$

if $b^* < (1+f^*)L^*$ or:

$$\hat{l}^* = 0;\ q^* = 0 \tag{12}$$

otherwise.

Proof: From Lemma 4 we know that if $b^* < (1+f^*)L^*$ then Σ_3 is a sink state, so that for any initial condition $(\hat{l}_0, q_0) \in X$ the state ends in Σ_3. Moreover, in Lemma 2 we have proved that Σ_3 has the asymptotic stable equilibrium (11), so that we can conclude that for any initial condition $(\hat{l}_0, q_0) \in X$ the state will asymptotically converge to (11).

If we assume $b^* > (1+f^*)L^*$, by following similar arguments, we can conclude that for any $(\hat{l}_0, q_0) \in X$ the state will asymptotically converge to (12). ■

Fig. 7 shows the trajectories obtained by considering initial conditions (marked with a $\times$) $(\hat{l}_0, q_0) \in X$. The figure shows that all the trajectories are attracted by the stable equilibrium $(\hat{l}^*, q^*)$ predicted by Proposition 1 and marked in Fig. 7 with a $\circ$.

Proposition 2: Under network congestion, i.e. $r^* > b^*$, the Skype controller produces a steady state overflow rate equal to:

$$o^* = \sqrt{b^* L^* (1+f^*)} - b^* \tag{13}$$

which means that Skype is not able to avoid congestion.

Proof: Being in Σ_3, Lemma 1 implies $b^* < (1+f^*)L^*$. From Proposition 1 we know that (11) is an asymptotically stable equilibrium for $\mathcal{H}$. Therefore, by considering (3), the steady state value of the overflow rate is given by:

$$o^* = (1-\hat{l}^*)L^*(1+f^*) - b^* \tag{14}$$

Now by substituting (11) in (14), it turns out (13). The overflow rate o^* is greater than zero since $b^* < L^*(1+f^*)$. In other terms, under congestion, the evolution of the system will be described by Σ_3 (see Lemma 4), thus the queue will be full ($q^* = q_M$) and the overflow rate will be persistent ($o^* > 0$). ■

In order to validate this Proposition through an experiment, a Skype flow has been injected in a bottleneck whose maximum queue length is $q_M = 10\,\text{KB}$ and that experiences an available bandwidth drop from 120 kb/s to $b^* = 30\,\text{kb}/s$ at $t = 50\,\text{s}$.

Fig. 8 shows the results of the experiment. The actual throughput, the loss rate and the queue length produced by the Skype flow are compared with the steady state values (represented in dashed thick lines in Fig. 8) predicted by the model. It is worth noting that the throughput at equilibrium ($r^* = (1-\hat{l}^*)L^* \simeq 40.988\,\text{kb/s}$) and the loss rate ($o^* = \sqrt{b^* L^*} - b^* \cong 10.988\,\text{kb/s}$) found by using Propositions 1 and 2 nicely match the steady state values measured through the experiment.

V. Conclusions

In this note we have proposed a hybrid automaton to model the congestion control algorithm implemented by the Skype VoIP application. Main findings are: 1) the loss ratio is the main driver of the Skype input rate; 2) the sending rate matches the available bandwidth with a finite error, so that Skype experiences a persistent congestion; 3) the sending rate matches the available bandwidth with a slow time constant.

References

[1] V. Jacobson, "Congestion avoidance and control," *ACM Comput. Commun. Rev.*, vol. 18, no. 4, pp. 314–329, 1988.

[2] L. Eggert and G. Fairhurst, "UDP Usage Guidelines for Application Designers," *RFC 5405*, Nov. 2008.

[3] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proc. of ACM SIGCOMM*, 2000.

[4] S. Low, F. Paganini, and J. Doyle, "Internet congestion control," *IEEE Control Systems Magazine*, vol. 22, no. 1, pp. 28–43, 2002.

[5] J. Lee, S. Bohacek, J. P. Hespanha, and K. Obraczka, "Modeling communication networks with hybrid systems," *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 630–643, 2007.

[6] S. Mascolo, "Congestion control in high-speed communication networks using the Smith principle," *Special Issue on "Control methods for communication networks" Automatica*, vol. 35, no. 12, pp. 1921–1935, 1999.

[7] C. Hollot, V. Misra, D. Towsley, and W. Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Transactions on Automatic Control*, vol. 47, no. 6, pp. 945–959, 2002.

[8] R. Srikant, *The Mathematics of Internet Congestion Control*. Birkhäuser, 2004.
[9] S. Guha, N. Daswani, and R. Jain, "An Experimental Study of the Skype Peer-to-Peer VoIP System," in *Proc. IPTPS '06*, Feb. 2006.
[10] S. Baset and H. Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol," in *Proc. of IEEE INFOCOM '06*, Apr. 2006.
[11] L. De Cicco, S. Mascolo, and V. Palmisano, "An Experimental Investigation of the Congestion Control Used by Skype VoIP," in *Proc. of 5th International Conference on Wired/Wireless Internet Communications (WWIC)*, Coimbra, Portugal, May 2007.
[12] ——, "Skype Video Responsiveness to Bandwidth Variations," in *Proc. of ACM NOSSDAV 2008*, Braunschweig, Germany, May 28–30, 2008.
[13] D. Bonfiglio, M. Mellia, M. Meo, D. Rossi, and P. Tofanelli, "Revealing skype traffic: when randomness plays with you," in *Proc. of ACM SIGCOMM '07*, Aug. 2007.
[14] L. De Cicco, S. Mascolo, and V. Palmisano, "A Mathematical Model of the Skype VoIP Congestion Control Algorithm," in *Proc. of IEEE Conference on Decision and Control '08*, Cancun, Mexico, 9-11 Dec 2008.
[15] W. Jiang and H. Schulzrinne, "Comparison and optimization of packet loss repair methods on voip perceived quality under bursty loss," in *Proc of NOSSDAV '02*, Miami, Florida, USA, 2002.
[16] S. Mascolo, "Modeling the Internet congestion control using a Smith controller with input shaping," *Control Engineering Practice*, vol. 14, no. 4, pp. 425–435, Apr. 2006.
[17] J. Lygeros, K. Johansson, S. Simic, J. Zhang, and S. Sastry, "Dynamical properties of hybrid automata," *IEEE Transactions on Automatic Control*, vol. 48, no. 1, pp. 2–17, 2003.
[18] "One-way transmission time," *ITU-T Recommendation G.114*, 1993.