Voice Adaptive Gateway Pacer for Wireless Multihop Networks

Dragoş Niculescu[1]
ETTI, University Politehnica of Bucharest

Kyungtae Kim, Sampath Rangarajan
NEC Laboratories America

Sangjin Hong
ECE, Stony Brook University

***Abstract*— When supporting both voice and TCP in a wireless multihop network, there are two conflicting goals: to protect the VoIP traffic, and to completely utilize the remaining capacity for TCP. We investigate the interaction between these two popular categories of traffic and find that conventional solution approaches, such as enhanced TCP variants, priority queues, bandwidth limitation, and traffic shaping do not always achieve the goals. TCP and VoIP traffic does not easily coexist because of TCP aggressiveness and data burstiness, and the self-interference nature of multihop traffic. We found that enhanced TCP variants (Reno, Vegas, C-TCP, CUBIC, Westwood) fail to coexist with VoIP in the wireless multihop scenarios. Surprisingly, even priority schemes, including those built into the MAC such as RTS/CTS or 802.11e, generally cannot protect voice, as they do not account for the interference outside communication range.**

We present *VAGP (Voice Adaptive Gateway Pacer)* - an adaptive bandwidth control algorithm at the access gateway, that dynamically paces wired-to-wireless TCP data flows based on VoIP traffic status. *VAGP* continuously monitors the quality of VoIP flows at the gateway and controls the bandwidth used by TCP flows before entering the wireless multihop. To also maintain utilization and TCP performance, *VAGP* employs TCP specific mechanisms that suppress certain retransmissions across the wireless multihop. Compared to previous proposals for improving TCP over wireless multihop, we show that *VAGP* retains the end-to-end semantics of TCP, does not require modifications of endpoints, and works in a variety of conditions: different TCP variants, multiple flows, internet delays, different patterns of interference, different multihop topologies, different traffic patterns.

I. Problem Statement

Most traffic that flows over the Internet makes use of the Transmission Control Protocol (TCP) and wireless multihop networks are one way to provide access extension. TCP is one of the protocols designed for wired networks and exhibits severe degradation in multihop networks. It was designed to provide reliable end-to-end delivery of data over unreliable networks and has been carefully optimized in the context of wired networks. For example, large TCP default window sizes that are appropriate for a wired network are too large for wireless links in multihop networks.

Another type of traffic that becomes more prevalent in homes and institutions is VoIP. This capability becomes available in most new cell-phones as well, due to convenience and cost savings. VoIP however is different from most other traffic in that it has quite stringent delivery requirements. While mechanisms to provide for this QoS exist in the wired networks, in the popular 802.11-based networks they were only an afterthought.

In this paper we show that coexistence between these two popular types of traffic is a difficult one in multihop networks, and investigate the different methods that can be used to facilitate it. Even if QoS enhancements such as 802.11e were added, it doesn't really address the central problem of multihop networks, which is interference, the main factor affecting the coexistence.

The interaction between TCP and VoIP over a multihop network is complex, here is a summary of most important points:

- *TCP is an end to end protocol.* There are no explicit signaling mechanisms in the network to tell the TCP peers how fast to send, how much to send, or when to slow down a transmission. A peer is responsible for controlling these parameters from implicit knowledge it obtains from the network or from explicit knowledge it receives from the other peer. TCP needs to be aggressive in discovering link bandwidth because this way it can achieve high utilization. This is achieved using large windows, which aggravates channel contention on wireless links.
- *TCP produces bursty traffic, while VoIP is uniform.* In the so called 'slow' start phase, TCP doubles its window for each ACK received - in reality an exponential increase in bandwidth consumption. This creates trains of packets that hog the medium for prolonged times. VoIP on the other hand needs regularity in the network delay and a low loss rate. When the network is congested by interference or too much TCP data, VoIP traffic suffers from increased network losses and delays. However, TCP just goes into the recovery stage, reducing its sending rate until the network is recovered from congestion, and then send all postponed packets. This cycle of burstiness leads to both low utilization for TCP and unacceptable quality for voice.
- *TCP assumes that losses come from congestion.* This observation has been the basis of many studies and proposed modifications focusing on preventing TCP congestion control mechanism to react to link layer errors. Many performance studies of the TCP protocol over 802.11-based multihop show standard TCP behavior may lead to poor performance because of packet drops due to hidden terminal induced problems such as channel interference

[1]partly based on results obtained while first author was employed by NEC Laboratories America.

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537

and TCP data/ACK contention.

- *VoIP packets are small, while TCP packets are large.* For a given bit error rate, TCP packets will have less success, so many of them would be retransmitted across multihop links, thus generating even more load that in turn generates more interference.

VoIP is mostly constant bit rate, has very tight delay and loss requirements, and should always be served prior to TCP traffic. Classical solutions such as priority queues, bandwidth limitation, and traffic shaping do not provide satisfactory solutions for the coexistence problem. Even if voice traffic has priority locally within a node, bursty TCP traffic affects voice packets on other nodes within the interference range.

This paper investigates the behavior of TCP and VoIP flows in a shared network and proposes a novel bandwidth control technique to enable this coexistence in interference-ridden conditions such as multihop networks and WLANs. We examine ways in which TCP and VoIP can coexist while **satisfying two contradicting goals**: maintenance of VoIP quality, but without sacrificing TCP performance and network utilization. We found that merely limiting bandwidth of TCP, while necessary, is not sufficient, as the bursty behavior of TCP results in poor utilization. Another finding is that it is beneficial to make TCP look more CBR like for two reasons: it is more predictable and therefore VoIP friendly, but also has **hidden benefits for TCP itself.**

The problem is not simply a matter of bandwidth estimation, even if VoIP takes a predictable share of the resources, because it matters not only how much TCP to allow in the network, but also what shape. We propose *Voice Adaptive Gateway Pacer (VAGP)* - a method that uses existing VoIP traffic that shapes incoming TCP to protect both VoIP and utilization. This estimation is a continuous process that has to adapt to a changing environment: wireless channel conditions, VoIP load, TCP flow arrivals, internet delays - all may change on time scales of seconds. *VAGP* limits TCP share within milliseconds of noticing reduced capacity, but still allows aggressive discovery when new bandwidth becomes suddenly available, all while protecting voice traffic.

II. Existing work

A large amount of research has focused on the optimization of the TCP performance over wireless networks. The majority of the solutions proposed by the research community fall in three main categories:

Connection splitting solutions: The key problem for TCP over hybrid wireless/wired networks lies in the different characteristics of wireless and wired networks. While most packet losses experienced in wireless networks are due to hidden terminals and channel contention at the intermediate nodes, drops in the Internet almost always are due to buffer overflows at the routers. An early solution to this network convergence problem [1] lies in splitting the TCP connection at the node interfacing the wired and wireless part of the network, denoted as the Internet gateway. Connection splitting can hide the wireless link entirely by terminating the TCP connection prior

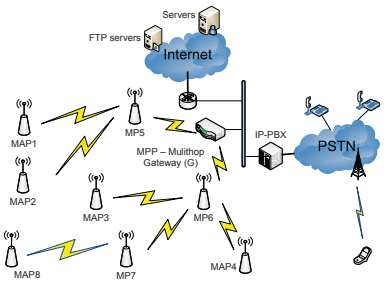


Fig. 1. Wireless multihop topology: a multihop gateway connects to the wired Internet to deliver TCP traffic, or to PSTN via IP-PBX for VoIP calls. A Multihop Point (MP) just forwards traffic, whereas a Multihop Access Point (MAP) also allows stations (STA) to associate with it. In this paper we consider downlink TCP traffic originating on the Internet, destined to a client associated with a MAP.

to the wireless link at the base station or access point. With this approach, the communication in the wireless part can be optimized independently of the TCP applications. However it requires extra overhead to maintain two connections for one TCP communication. It also violates end-to-end semantics and complicates the handover process.

Link layer solutions: These try to make the wireless link layer look similar a wired layer from the perspective of TCP. The most relevant and interesting proposal is the snoop protocol [2]. A snoop agent is introduced at the base station to perform local retransmissions using information sniffed from the TCP traffic passing through the base station. Another link layer solution proposes QoS scheduling with priority queues in the access point (AP) [3] to improve VoIP quality by placing TCP data in a lower QoS level.

Gateway solutions: One way to address TCP performance problems within wireless networks is to evenly space, or pace data sent into the multihop over an entire round-trip time, so that data is not sent in a burst. Pacing [4] [5] can be implemented using a data and/or ACK pacing mechanism. TCP-GAP [4] suggested congestion control scheme to reduce burst of TCP packets based on estimating 4-hop propagation delay and variance of recent RTTs at the Internet gateway for wired-wireless hybrid networks. TCP-GAP scheme is relatively responsive, provides fairness among multiple TCP flows and shows better goodput than TCP-NewReno. However, it still depends on network topology, and fails to estimate TCP bandwidth correctly in the presence of real-time traffic, such as VoIP. Congestion control is still operated by the traditional TCP scheme, which is too aggressive for wireless multihops.

III. TCP and VoIP: Difficult coexistence

It is well understood from queuing theory that bursty traffic produces higher queueing delays, more packet losses, and lower goodput. It has been observed that TCP's congestion control mechanisms and self-clocking create extremely bursty traffic in networks with large bandwidth-delay products and cause long queues and the likelihood of massive losses. In addition, wireless multihop network traffic tends to have a self-similar behavior [6] making it difficult to provide stable rates necessary for VoIP.

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537

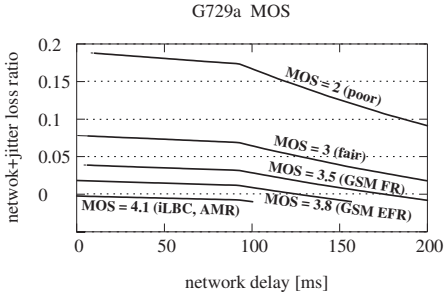
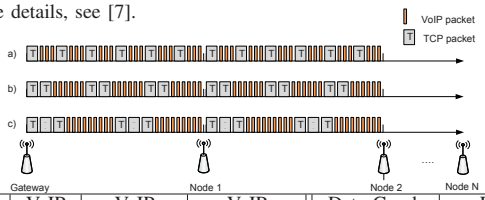


Fig. 2. For certain vocoders, such as G729a, VoIP quality (MOS-$score$) can be computed as a function of loss and one way delay. Loss include packets lost in the network, and packets which miss their deadline because of jitter. For more details, see [7].



burst length	VoIP calls	VoIP Loss (%)	VoIP delay (ms)	Data Goodput (Kbps)	Data delay (ms)
1	5	0.92	13	509	15
2	5	1.26	16	501	19
3	4	1.44	17	495	23
4	4	1.64	19	488	26
5	3	2.06	21	476	31

Fig. 3. VoIP statistics and data goodput as the burstiness increases; 5 VoIP calls and 550Kbps of data offered; 4-hops string topology, 12Mbps, 802.11a.

Figure 1 shows the wired/wireless hybrid networks we consider in this paper. A multihop extension carries traffic from wired Internet, or PSTN (through IP-PBX). The multihop leg is where VoIP needs to be protected from TCP.

A. Bursty traffic

To understand the difficulties in supporting VoIP, we start with a short primer on VoIP quality requirements. VoIP traffic is CBR, and for certain vocoders (G711, G729), its quality can be estimated using only packet loss and mouth-to-ear (one way) delay. Figure 2 shows the values of MOS-$score$ with respect to network delay and total loss for 60ms playout-buffer and 25ms vocoder delay. In order to obtain MOS-$score$ of 3.6 (comparable to GSM quality), the network has to deliver all packets in less than 160ms, or deliver 98% in less than 104ms. For G.729a used in the rest of the paper, 3.9 is the maximum quality achievable, but we consider 3.6 to be acceptable quality (roughly similar to GSM).

First, we show the burstiness is the main cause of reduced VoIP quality. To this end, we experiment with various packet patterns as shown in Figure 3 top. In these scenarios we have the same mean offered rate for data (550Kbps), but with different burst lengths. The rest of the capacity is filled by VoIP packets.

The results corresponding to different burst lengths are shown in the table in Figure 3. Virtually all quality indicators for both VoIP (loss, one-way delay) and data (goodput, one-way delay) suffer because of the increased burst length. In fact, we can support 5 voice calls with 1 data packet bursts, but only 3 with 5 data packet bursts. In Internet scenarios, when long delays can be present on the Internet portion, even one TCP flow is expected to require windows much larger than 5 packets, and therefore produce even more degradation for itself and for VoIP.

In the same topology of four hops we try to establish what kind of performance we can expect from each type of traffic, and in combination. Running each of the four hops at 12Mbps, we can either support 11 VoIP calls, or 1.35Mbps of TCP. However, if we mix 5 VoIP calls and 3 TCP flows, we found that voice quality is below the minimum acceptable ($MOS < 2$) while TCP flows get a cumulative goodput[2] of 615Kbps using a throughput[3] of 903Kbps. **This shows that simply sharing the network fails to protect the VoIP traffic, and also yields lower utilization.** TCP uses a sliding window-based protocol which determines the number of packets that can be sent, and uses the receipt of acknowledgments to trigger the sending of packets. The window used by a TCP sender is chosen based on its view of the congestion in the network and based on the receiver's acceptable number of bytes. If the window size is too large, then the sender is allowed to inject traffic more than the network can handle. Given a wireless multihop network, there exists a TCP window size W^* at which TCP's bandwidth consumption is appropriate. This number depends on many conditions, including presence of real time traffic, but the main point is that default TCP algorithms are not able to discover this W^*. The current TCP protocols do not operate around W^* but instead typically grow their average window much larger. This results in VoIP degradation, or in low TCP performance if VoIP traffic is not present [8].

B. Behavior with mixed traffic

Continuing with the four hop scenario, we then mix these two types of traffic: five voice calls are started at the beginning of the experiment, and after 10 seconds three TCP flows are started in parallel with the voice. Figure 4 shows how VoIP quality is degraded as TCP window size increases. VoIP quality is shown in the low-right sub-figure, which starts at MOS-$score$ of around 3.9 and decreases to MOS-$score$ of around 2.2 as soon as the TCP flows with default window size 32 are introduced. TCP flows initially go through the "slow-start" phase increasing the number of packets in transit exponentially, which means every ACK packet triggers twice as much data as it acknowledges. After reaching the congestion threshold, TCP switches to linear rate increase to maintain high goodput without causing congestion in the congestion avoidance phase. These two alternating phases produce the well known saw-tooth pattern shown in upper left sub-Figure 4. In the lower-left sub-Figure, we see an additional disadvantage that is wireless specific: the self-interfering nature of the wireless multihop, coupled with well known 802.11 unfairness issues [9] causes severe unfairness between the TCP flows. At

[2]goodput = total bytes delivered by TCP receiver to the application layer.
[3]throughput = total bytes sent by the TCP sender.

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537

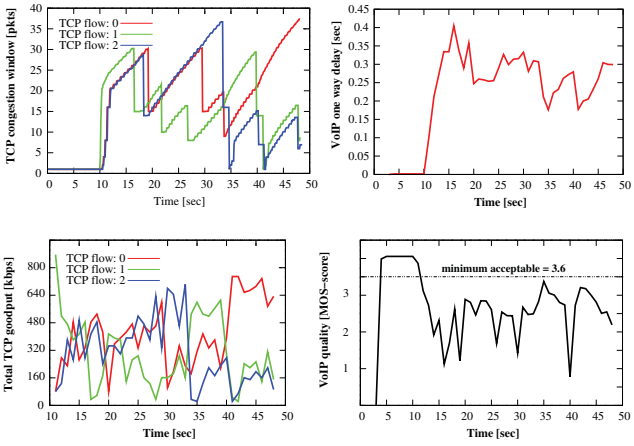


Fig. 4. Uncontrolled TCP has many drawbacks: built-in backoff mechanism of TCP reacts too late to protect VoIP; Increased one-way network delay for VoIP; unfairness between TCP flows; low total utilization.

the top right of Figure 4, we trace the delays experienced by VoIP packets of one of the flows during the same experiment: due to TCP packet bursts and large packet size, network delay jumps from 10ms to about 250ms mostly due to extra wait in the queues. This is the main driver of reduced voice quality, as large queues and 802.11 retries largely cover most network losses in this example.

IV. Candidate Solutions

It is clear from the previous section that VoIP and TCP cannot simply share a multihop network without experiencing severe reduction in TCP capacity and voice quality degradation. We first consider the various enhancements to TCP proposed by the research community in the recent years, namely: Reno, Vegas, Westwood, CUBIC, and Compound TCP (C-TCP). Then, we look at external control methods, that would leave TCP unchanged, but would police or instrument traffic at the multihop gateway.

A. TCP variants

TCP Reno [10] is the most widely deployed algorithm. It induces packet losses to estimate the available bandwidth in the network. With the additive increase and multiplicative decrease mechanism, it causes a periodic oscillation in the window size which results in large delay jitter and bursty packets. *TCP Vegas* [11] was introduced with the idea that it is more efficient to prevent congestion than to fix it. The modified mechanisms use observed delay to detect an incipient stage of congestion and try to adjust the congestion window size before packets are lost. The key innovative idea of *TCP Westwood* [12] is to continuously measure the packet rate of the connection at the TCP sender by monitoring the rate of returning ACKs while trying to find the bandwidth estimate which is defined as the share of bottleneck bandwidth available to the connection. With the idea that pure loss-based or delay-based congestion control approaches that improve TCP goodput in high-speed networks may not work well, the following algorithms are designed to combine two approaches. *C-TCP* [13] can rapidly increase sending rate when network path is under utilized, but gracefully retreat in a busy network when bottleneck queue grows. *TCP CUBIC* [14] was proposed to address the under-utilization problem due to the slow growth of TCP congestion window in high-speed networks. *C-TCP* is included in Windows Vista, and *CUBIC* in the newest Linux kernels, so these TCP variants are likely to see wide deployment in the coming years.

What is true for all TCP variants is that data packets arrive at the receiving host at the rate that the bottleneck link will support. A TCP sender's self-clocking depends on the time spacing of ACKs being preserved end to end. Jitter introduced by in network queues misleads the sender into pushing more data than the network can accept. Cumulative acknowledgement or ACK compression may cancel the spacing of the ACKs and result in bursty traffic with a high risk of high peak rate beyond network capacity. A single acknowledgement can acknowledge several thousands of packets, opening up the window in a large burst.

We compared these five TCP variants with respect to their capacity to coexist with VoIP and utilization of the multihop. Figure 5 shows that all TCP variants fail to protect VoIP in a simple shared environment. What they do is to increase TCP goodput with large window size. Vegas exhibits both a better VoIP protection and utilization of the multihop links, due to its balanced congestion control in low number of VoIP flows. Surprisingly, Westwood, which is designed specifically for lossy links performs worst on both measures, wasting half of the capacity on retransmissions $\frac{goodput}{total\_sent} \approx 0.5$ (not shown in the figure).

B. Policing TCP traffic

In fact, an even more likely situation is that none of the TCP endpoints can be controlled because upgrading TCP is unfeasible or undesirable for other reasons. Even enhanced TCP endpoints cannot possibly protect wireless multihop networks in the path. We therefore explore other methods to enable the coexistence at the gateway into the wireless multihop. Policing of TCP traffic can be performed using classical methods such as priority queues and traffic shaping, or by instrumenting TCP packets to manipulate receiver's advertised window (awnd). Any of these methods has the goal of reducing the amount or the shape of the TCP data pushed into the multihop.

1) Priority queues: One solution to harmonize VoIP and TCP traffic is the use of priority queues. We simulated priority queues in ns-2 allocating the highest priority for VoIP traffic in all nodes. We found that only 20% of the voice capacity can be used, and only for one or two hops. For cases of three or more hops, priority queues are not able to support any amount of VoIP traffic. The reason is that priority queues, or even 802.11e[4] cannot protect from interference generated two or three hops away. On the contrary, this approach even increases packet burstiness while building up TCP packets in the queue. These localized approaches cannot provide solution to a global problem of hidden terminals interfering across several hops.

[4]802.11e uses multiple queues for downlink traffic, and preferential contention parameters for uplink traffic in order to offer priority to QoS traffic.

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537

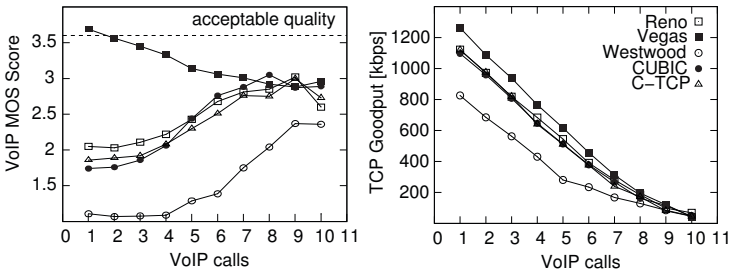


Fig. 5. Left: VoIP quality with different variants of TCP; Right: TCP goodput. TCP variants are too aggressive and use all available bandwidth, reducing voice quality. However, with more voice calls TCP experiences more packet loss, which leads to retransmissions and frequent slow start phases.

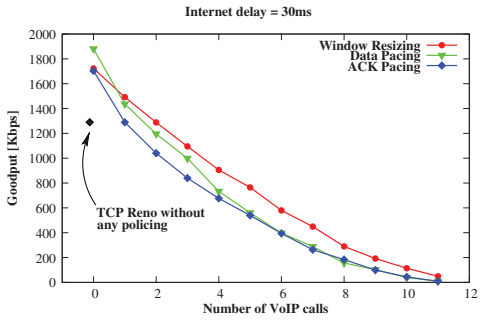


Fig. 6. Shared capacity with VoIP and TCP using data pacing, ACK pacing, and window control. All methods are bounded by the nominal capacity of the network. Window resizing reduces total number of retransmissions, which leads to a higher TCP goodput compared to data/ACK pacing

2) Window resizing: TCP bandwidth discovery operates from the sender, and cannot be easily manipulated. The advertised window of the receiver however, can be instrumented in the network to reflect the actual bandwidth available in the wireless network. In concordance with previous studies, we found that limiting TCP sending behavior has beneficial effects even in the case when only TCP traffic is in the network. In order to control TCP sending rate without modification of TCP endpoints and maintain end-to-end semantics, we modify the advertisement window in each ACK packet at the gateway. This method limits the total number of TCP data packets in transit between the end points. If the gateway changes TCP advertisement window based on the network status, TCP throughput can be limited close to its entry point. By keeping the window size small to protect VoIP, retransmission as well as fairness problems among TCP flows are also relieved.

We experimented with various values for the advertised window and found that smaller windows are more beneficial than larger windows. This is a consequence of the fact that TCP's share of the wireless medium needs to be reduced [8].

3) TCP data/ACK Pacing: One problem that is not solved by window resizing is that of packet bursts. TCP pacing promises to reduce burstiness of TCP traffic and alleviate the impact of packet loss, network delay, and delay jitter of VoIP traffic. TCP pacing evens out the transmission of a window of packets based on a shaper parameter R. After a packet of size pkt_size goes out in the air, the next packet is scheduled no earlier than $\frac{pkt\_size}{R}$. The gateway chooses a rate R based on the network status to determine how much to send as well as when to send. One way to understand the impact of pacing is to consider burstiness from network delay, jitter and packet loss perspective. With bursty traffic, packets arrive all at once at the gateway. As a result, queueing delay and delay jitter of VoIP packets grows linearly with TCP load due to large packet size, even when the load is below capacity. From the viewpoint of TCP, 802.11 links which support VoIP traffic still have large capacity for TCP data; ignoring the interference side effects that are felt several hops away from the link in question and reserved but unused bandwidth for the delay and jitter compensation for VoIP.

In our measurements, the pacer offered protection to VoIP at the cost of sacrificing available bandwidth for retransmissions. While providing benefits such as small buffer size at the pacer, ACK pacing may fail to prevent bursty data packets which results in low TCP performance and degradation of VoIP quality. The disadvantage of pacing is that buffer overflows at the gateway due to capacity fluctuation will cause packet drops and increase queueing delay. The increased queueing delay easily causes TCP retransmission timer to expire, which results in retransmitting the packets already transferred to the receiver, unlike the window resizing solution. However, the main advantage is that it works with higher number of hops, and does not require instrumentation of TCP packets.

In Figure 6, we look at how TCP and VoIP can share the available bandwidth using window control and data/ACK pacing. On the horizontal axis, we increased the number of calls from 1 to 11, and attempted to maximize the TCP goodput while still maintaining $MOS\text{-}score$ of 3.6 for the VoIP traffic. While all three control methods achieve some amount of sharing between the two types of traffic, the window control attains better utilization. The benefit of TCP policing is visible even without VoIP traffic when plain TCP wastes capacity on retransmissions achieving a lower goodput.

The methods examined have a straightforward application only when the wireless capacity is fixed. In reality, the capacity is highly variable depending on a multitude of factors: number of hops and their configuration, the amount and type of interference, the actual capacity of each hop, the amount of voice to be served. To support VoIP under varying conditions, the basic policing tools examined here should be used in conjunction with methods to dynamically estimate available bandwidth in real-time.

V. *Voice Adaptive Gateway Pacer*

Having established that pacing is an appropriate method to control TCP in a shared network in Section IV-B.3, the question is what data rate should TCP get? We can divide bandwidth statically between the voice calls and the amount of TCP, but the static bandwidth sharing causes poor network utilization when usage is low for TCP and high for VoIP or even fails to protect voice in case of poor wireless link. In reality, the capacity is highly variable depending on a multitude of factors: network topology and their configuration, the amount and type of traffic, the actual capacity of each

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537

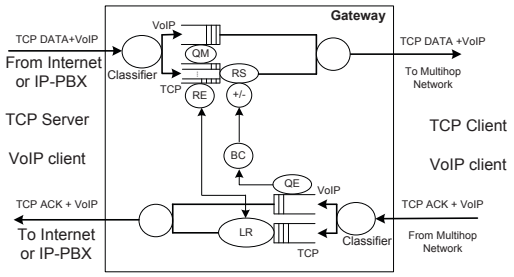


Fig. 7. *VAGP* functionality installed in the gateway monitors both VoIP and TCP traffic. It controls TCP sending rate based on the current estimation of VoIP quality. QM: Queue Manager, RS: Rate Shaper, RE: Redundant Eliminator, BC: Bandwidth Controller, QE: VoIP Quality Evaluator, LR: Local Recovery

node, the amount of voice and other traffic to be served for each node. *Voice Adaptive Gateway Pacer*, uses the existing voice traffic as probe traffic to estimate the rate that can be given to TCP:

$$R_{TCP} = \sum_{i=1}^{N} R_{TCPi} = \left(\frac{1}{\widehat{SRTT}} + R_{adjustment} \right). \quad (1)$$

The bandwidth for TCP flows can be estimated by measuring the average of recent RTT samples, $\widehat{SRTT}$, with the rate adjustment, $R_{adjustment}$, from the variation of VoIP quality. The adaptive transmission rate R_{TCP} for TCP bandwidth computed by the multihop gateway is allocated to each TCP flows following the bandwidth sharing policy.

In order to smooth out TCP burstiness either for flows with a large BDP[5] which have intra-flow burstiness, or for other flows which may show inter-flow burstiness, rate based traffic control is necessary. *VAGP* is designed to reduce TCP packet bursts by releasing packets smoothly into the network rather than in bursts. With long end-to-end delays, TCP application tends to inject enough packets to fill high BDP paths [15]. With wireless multihops this creates congestion, and drops in the multihop portion require retransmissions, which reduce overall utilization. The method we propose controls the data transmission rate and dynamically adjusts the parameters for the actual TCP sending rate to consume residual bandwidth while keeping good VoIP quality. *VAGP* has two main functions: to tame the bandwidth discovery nature of the TCP at least for the wireless multihop portion making the TCP friendly to VoIP traffic, and to maintain high utilization.

A. Basic idea

At the gateway shown in Figure 7, VoIP Quality Evaluator (QE) monitors the network parameters of VoIP traffic, including network delay, network loss, and jitter loss. QE calculates $MOS\text{-}score$ of the monitored VoIP and reports it to Bandwidth Controller (BC). Based on the $MOS\text{-}score$, BC estimates the bandwidth portion of TCP traffic, translates the available bandwidth into TCP sending rate and informs TCP Rate Shaper (RS) of TCP data rate at regular period D_{normal}.

[5]Bandwidth×delay product (BDP) refers to the product of a data link's capacity and its end-to-end delay (sometimes the data link's capacity times its round trip time).

RS distributes available bandwidth to each TCP flows. Algorithm 1 briefly shows how *VAGP* works. It first estimates the bandwidth portion for TCP traffic with the measured $MOS\text{-}score$ which is classified using four thresholds for triggering TCP rate adjustment such as Q_{good}, Q_{fair}, Q_{poor}, Q_{choke}. The parameters for TCP rate adjustment are increasing the bandwidth by R_{good}, R_{fair} or decreasing it by R_{poor}, R_{choke}.

The algorithm is executed every $period$ for stable VoIP one-way network delay D_{normal}, but if the network delay is larger than the threshold D_{alert}, QE triggers more often using a shorter $period$. As shown in Figure 2, an $MOS\text{-}score$ of 3.6 requires loss less than 2% and delivery in less than 170ms. Due to reduced capacity or higher load, one way VoIP network delay can reach the threshold D_{alert} for a given 2% packet loss ratio. In this situation, normal TCP rate adjusting $period$, P_{normal}, is not enough to react quickly. Thus, QE reduces $period$ to P_{alert}.

Algorithm 1 Bandwidth Controller (BC)

```
Voice quality parameters: Q_good, Q_fair, Q_poor, Q_choke;
TCP rate parameters: R_good, R_fair, R_poor, R_choke;
Input: continuous VoIP traffic
Output: R, R_i TCP shaper rates
R_i ⇐ Distribute R to each TCP flow, for example fair share.
D_voip: one way VoIP network delay
P_normal: rate adjustment period for D_normal, 100ms
P_alert: rate adjustment period for D_alert, 20ms
loop
  for all TCP flow i do
    if MOS-score > Q_good then
      R_i ⇐ R_i(1 + R_good);
    else if MOS-score ∈ [Q_good, Q_fair) then
      R_i ⇐ R_i(1 + R_fair);
    else if MOS-score ∈ [Q_fair, Q_poor) then
      R_i ⇐ R_i(1 − R_poor);
    else if MOS-score ≤ Q_poor then
      R_i ⇐ R_i(1 − R_choke);
    end if
  end for
  if D_voip < D_alert then
    adjust period ⇐ P_normal;
  else if D_voip ≥ D_alert then
    adjust period ⇐ P_alert;
  end if
  wait(period) while evaluating VoIP quality;
end loop
```

B. Quick Responsiveness

For quick responsiveness of the rate adjustment, RS needs to keep the highest achieved rate, R_{max} for TCP bandwidth while $MOS\text{-}score$ is larger than Q_{good}. R_{max} will be utilized to react quickly to reach the optimum share for TCP bandwidth. If voice quality is recovered to Q_{good} from Q_{poor} and the good voice quality is held for some period i.e. 5 x $period$, RS makes use of R_{max} to compare the estimated R_i with half of R_{max}. If the estimated R_i is less than $\frac{R_{max\_rate}}{2}$, RS increases R_i to $\frac{R_{max}}{2}$. In the normal operation of R_{good}, there is a possibility of non-work-conserving transmission, that is, TCP flows experience long period of low bandwidth utilization before it reaches to the maximum share. Through this mechanism, *VAGP* swiftly can recover optimum TCP share R while protecting VoIP. The experimentation result is shown in V-E.

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537

C. Local Recovery and Redundant Packet Drop

In Algorithm 1, BC limits TCP share of the bandwidth in the favor of VoIP traffic. However, it may cause TCP performance degradation due to queue overflow at the gateway or TCP timer expiration from the sender with the larger queueing delay. TCP timer expiration results in the multiple packet retransmissions again from the lost packet if TCP sender adopts Go-back-N algorithm to recover the packet loss. Due to larger packet size, TCP suffers higher packet drop ratio over interference-ridden multihop environment. The packet drop decreases the sending rate at the sender side, which results in poor performance in high BDP networks. Also it may trigger TCP retransmission timer to expire which causes the packets already arrived at the destination to be sent again over the wireless network. This wastes the multihop resource and decrease TCP performance even more.

To fully utilize the space left by voice traffic, *VAGP* uses two mechanisms: 1. local retransmission, 2. redundant retransmission drop. Algorithm 2 outlines the functionality provided by these two procedures. When a duplicate ACK is received, Local Recovery (LR) retransmits the lost packet to the destination eliminating the delay of the packet transmission from the sender. The packets with lower sequence number lower than or equal to $highest\_seq$ are dropped at the Redundant Eliminator (RE) to prevent unnecessary packet transmission because they are already reached to the destination.

Algorithm 2 Voice Adaptive Gateway Pacer (VAGP)

```
TCP packet sequence number: seq
TCP highest acknowledged sequence number: highest_seq
PQueue: Pacing Queue of TCP Rate Shaper
loop
  for all received packets do
    if packet type == VoIP then
      send to QE;
    else if packet type == TCP Data then
      read seq from TCP header
      if seq <= highest_seq then
        Drop it (Redundant packets elimination);
      else
        send it to PQueue
      end if
    else if packet type == TCP ACK then
      read seq from TCP ACK header;
      if duplicated ACK then
        retransmit TCP data with seq + 1 (Local recovery);
      else
        update highest_seq
      end if
    end if
  end for
end loop
```

D. Queue Management

Instead of transmitting packets immediately upon receipt of TCP data, RS may delay transmitting packets to spread them out at the rate controlled by BC, causing a increased queueing delay, which results in long RTT at the TCP sender. Queueing delay increases linearly until the number of packets reaches to min($awnd$, $cwnd$). The local recovery mechanism of *VAGP* requires more delay while the lost packets in multihop are recovered by local retransmission. With no packet drops and little retransmission timer expiration, TCP sender may put more packets considering the multihop as a large bandwidth-delay product network, which increase queuing delay further. To address these problems, we propose an adaptive queue-aware window control algorithm to minimize the queuing delay and buffer requirement at the gateway. Let us first consider TCP window size. Suppose the multihop extension is bottleneck in hybrid wired/wireless network. Thus the queueing and network delay in multihop is much larger than the one in the wired part. From these assumption, one can easily derive the following condition.

$$q_i(t) + M_i(t) < min(awnd(t), cwnd(t)) \quad (2)$$

At an arbitrary time t, let us denote the queue length of RS as $q_i(t)$ and the number of packets in delivery over the multihop (max_sent_seq - $highest\_ack$) as M_i. We use subscript i to index the connections. Eq. 2 represents the maximum number of packets in transit which are clearly smaller than TCP sender's window size. One can also see the queue length can be controlled by modifying the advertisement window (awnd) in the ACK packets. In the queue-aware window control management, we use the downstream queue length $q_i(t)$ to represent the queueing delay. The basic control strategy for the queue management is as follows: 1. Queue Manager (QM) reduces $awnd$ to $q_{min} + M_i$ as soon as the queue length exceeds a threshold, q_{max_i}; 2. QM increases $awnd$ to $q_{max\_i} + M_i$ when the queue length falls below a threshold q_{min_i}, here $q_{min\_i} < q_{max\_i}$; 3. In case of 1, if allocated TCP bandwidth is 0, QM assigns 0 to $awnd$ to freeze all timers at the TCP sender. Otherwise, $awnd$ will be set to $\frac{q_{max\_i}+q_{min\_i}}{2} + M_i(t)$. We use q_{min_i} = 2 and q_{max_i} = 6 which is a reasonable choice to obtain low queueing delay while preventing buffer underflow. This strategy is shown to reduce the queueing delay while keeping minimum number of packets - low queueing delay, as well as preventing the underflow of the queue. Our in-depth simulation results have shown reduction of RTT from 3 seconds to 310 msec in case of 4 hops in Figure 8 while the same TCP goodput and voice quality are achieved as the one without queue management strategy.

Algorithm 3 Queue-aware window control operation in QM

```
if q_i(t) < q_min_i then
  awnd ← q_max_i + M_i(t)
else if q_i(t) > q_max_i and R_i(t) > 0 then
  awnd ← q_min_i + M_i(t)
else if q_i(t) > q_max_i and R_i(t) == 0 then
  awnd ← 0
else
  awnd ← (q_max_i + q_min_i)/2 + M_i(t)
end if
```

E. VAGP evaluation

In this section we evaluate *VAGP* for the voice quality achieved, TCP goodput (data rate seen by applications), and TCP throughput (data rate used by TCP). The difference

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537

hop cnt	Reno		TCP-GAP		VAGP	
	VoIP	TCP	VoIP	TCP	VoIP	TCP
1	0.9	2596 2654	3.9	2587 2645	3.9	2587 2634
2	1.3	1070 1312	3.1	997 1189	3.6	750 791
3	1.9	552 869	2.3	545 853	3.7	664 705
4	2.5	309 558	3.3	287 509	3.7	325 374

Fig. 8. Average VoIP quality (MOS), TCP goodput and throughput(Kbps) with the 70% VoIP traffic and 30% TCP; Internet delay = 30ms.

between the latter two is due to losses in the wireless multihop which lead to TCP retransmissions.

String Topology: we first consider the simple string topology with both TCP and VoIP being transported across the same four wireless hops through the gateway G in Figure 1. The TCP data originates across the Internet, on a node labeled $Server$, and is sent a client attached to the access point labeled $MAP8$. VoIP traffic may originate in the PSTN network, but travels across the same four wireless hops through the enterprise IP-PBX to/from another client attached to $MAP8$. *VAGP* functionality is added at the gateway node G. All the wireless links operate with 802.11a at 12Mbps. This is a good example of multihop connectivity in enterprise networks which use PSTN/IP-PBX for VoIP traffic and Internet for TCP traffic. G.729 codec produces 50 packets per second of 20 bytes each in each direction. We measure all the VoIP characteristics (delay, jitter and packet loss) contributing to $MOS\text{-}score$, and TCP goodput achieved using *Voice Adaptive Gateway Pacer*.

QE monitors VoIP quality of the voice traffic from G which is the entry point of the TCP flows. It reports the $MOS\text{-}score$ to Bandwidth Controller (B) at some $period$. For these experiments, the parameter values which are used are Q_{good}=3.9, Q_{fair}=3.7, Q_{poor}=3.6, Q_{choke}=3.3 for VoIP quality and R_{good}=5%, R_{fair}=3%, R_{poor}=10%, R_{choke}=50% for TCP rate adjustment. These values are somewhat conservative to preserve voice quality, as drops are inevitable at sudden changes in channel conditions or traffic patterns. But, as we will show in the next experiments these values work across widely different patterns and conditions.

In Figure 8, we see that *VAGP* significantly outperforms TCP-GAP with respect to the basic premise of isolating the voice traffic. Using Reno, TCP traffic occupies the entire available bandwidth at cost of VoIP quality, while both flows in TCP-GAP share the available bandwidth giving a larger portion of the bandwidth to TCP traffic which results in poor voice quality. In case of 4 hops running 7 voice calls and 3 TCP flows, *VAGP* achieves the fair share - $MOS\text{-}score$ = 3.7 and 325Kbps TCP goodput, while TCP-GAP allocates more bandwidth to TCP, which results in poor voice quality, $MOS\text{-}score$ = 3.3. In fact, *VAGP* also achieves a higher aggregate goodput than TCP Reno and TCP-GAP. Thus, more number of VoIP calls can be supported using *VAGP*, with TCP getting an optimum share of the bandwidth.

With the design to operate over 4-hops, TCP-GAP shows poor performance within less than 4-hop topology. At 4 hops, *VAGP* produces a slight increase in RTT, but with a lower standard deviation:

In summary, for the string topology *VAGP* improves **both** VoIP performance and network utilization ($\frac{goodput}{throughput}$).

Mixed flow sources/destinations: We consider the case

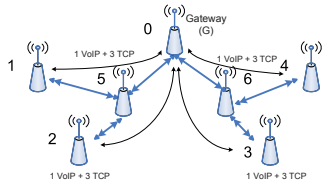


Fig. 9. Tree topology: downstream TCP flows are set up between wired servers (not shown) and leafs. VoIP sessions are set up between IP-PBX (not shown) and each leaf. Performance shown in Figure 10.

RTT (ms)	Reno	TCP-GAP	*VAGP*
mean	242	288	312
std	271	209	100

where three VoIP flows and one TCP flow originate in the wired domain and end at MP6, MAP3, MP7, and MAP8 respectively:

Topology	TCP-GAP		VAGP	
	VoIP	TCP	VoIP	TCP
String - 4 hops	2.6	437 732	3.7	309 346
Tree - 2 depth	1.1	985 1125	3.8	416 463

Fig. 10. *VAGP* performance compared with TCP-GAP with mixed flow destinations: string and tree. Columns show MOS score for voice, goodput and throughput for TCP. *VAGP* achieves VoIP protection as well as TCP performance with little sacrifice of the aggregate goodput. Standard TCP is not included because it doesn't perform in the simple string case.

Tree Topology: as a more complex topology, we consider a tree which consists of seven nodes placed in a tree two hops deep, with the gateway G is positioned at the root of the tree as shown in Figure 9. When run separately, this topology can support either 2 VoIP calls per leaf, or 3199 Kbps of aggregate TCP goodput in case of 3 TCP flows per leaf node. If we mix 1 VoIP call and 3 TCP flows for each leaf, *VAGP* gets less goodput than TCP-GAP, but VoIP quality is maintained above the $MOS\text{-}score$=3.6 required (second row in Figure 10).

***VAGP* with TCP variants:** we consider the same string topology as depicted in Figure 1 to show fairness between TCP variants under the control of *VAGP*. Running 5 TCP flows consisting of five TCP variants between wired server and MAP8 and 5 VoIP calls between PSTN and MAP8, we see that *VAGP* works well with any type of TCP. A fair share of around 20% is maintained between TCP flows while 5 VoIP calls are supported with an acceptable quality of $MOS\text{-}score$ = 3.9:

Hop count	VoIP (MOS)	Total TCP 627 (Kbps)				
		RENO	C-TCP	CUBIC	VEGAS	WW
4	3.8	125 (19%)	126 (20%)	126 (20%)	124 (19%)	126 (20%)

Dynamic Bandwidth Estimation: in this experiment using the same string topology as depicted in Figure 1, we verify the capability of *VAGP* to support VoIP together with TCP flows when the actual capacity of the multihop link is varying. The timeline in Figure 11 shows how *VAGP* adjusts TCP bandwidth consumption for one TCP flow while keeping good VoIP quality for 5 VoIP calls when the capacity of a 802.11a link fluctuates. On the horizontal axis, we have time in seconds, the top graph shows the voice quality, while the bottom graph shows the TCP end to end goodput. The bitrate of the link between nodes MP6 and MP7 is changed in the

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537

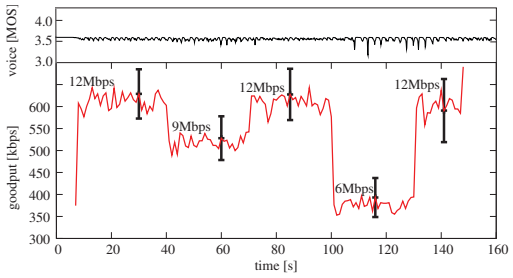


Fig. 11. TCP uses the residual bandwidth while VoIP quality is kept high, 5 calls 1 TCP over 4 hop string topology, G729a, 12Mbps, link capacity between MP6 and MP7 varies from 12 Mbps to 9 Mbps during 40s - 70s, 12 Mbps during 70s - 100s, 6 Mbps during 100s - 130s, 12 Mbps during 130s - 150s. The vertical bars represents the standard deviation of TCP goodput

following sequence: 12Mbps-9Mbps-12Mbps-6Mbps-12Mbps at times 40s, 70s, 100s, 130s. This emulates the behavior of a rate adaptation algorithm, or simply the variation in capacity of an indoor wireless channel. VoIP maintains average $MOS\text{-}score$ above 3.6 for all calls during this period.

Quick Responsiveness: looking at the detail of the timeline around 100s, we can see how *VAGP* swiftly reduces the TCP share R in order to protect VoIP. After an initial drop of R_{choke}=50%, R is adjusted up using R_{good}=5% to the feasible rate of around 0.38Mbps. This event is completed in 4s, ensuring that VoIP quality is maintained.

Different interference scenarios: all the simulations so far assumed a worse case interference scenario when hidden terminal relationships predominate both in the tree and the string. We also experimented with carrier sense dominated scenarios, when *VAGP* tends to perform better, but we do not include these results due to lack of space.

VI. Summary

TCP and VoIP can coexist in interference-ridden multihop networks, but only with some policing help. We evaluated a number of possible solutions: TCP variants, priority queues, MAC support. None is able to protect VoIP and to produce good utilization, as TCP wastes wireless capacity on retransmissions. We proposed *VAGP*, a method that addresses **both goals** of VoIP protection and network utilization. It uses existing VoIP traffic to continuously estimate the amount of bandwidth that can be dedicated to TCP. Making TCP more CBR like is beneficial in two respects: it becomes more VoIP friendly, but also minimizes self interference, which is beneficial for TCP itself, as it suffers disproportionately because of its large packets.

VAGP is shown to have the following advantages: 1. can be placed in the wireless gateway to monitor downlink TCP traffic; 2. maintains end to end semantics and compatibility; 3. doesn't require setting of parameters. An initial setting regarding the voice quality desired works across different conditions and situations. 4. provides complete separation, good utilization, and swift responsiveness to changing conditions; 5. performs well in a variety of conditions - various network topologies, several TCP flows, diverse network delays, different interference patterns, varying wireless capacity.

Currently we are developing a multihop solution for upstream TCP traffic originating in the wireless domain. Even if traffic is regulated at the MAP, the aggregation at MPs still produces burstiness, which calls for distributing some of the *VAGP* functionality in the MPs. Another interesting development direction is for the mesh topology. In this case, traffic policing needs to be done in a concerted manner across all gateways, considering traffic across the entire mesh.

Acknowledgment: This work was supported in part by CNCSIS grant PN2 Resurse umane 11/01.07.2009, and by FP7 grant SMART-Net 223937.

References

[1] A. Bakre and B.R. Badrinath. I-TCP: indirect TCP for mobile hosts. *Distributed Computing Systems, 1995., Proceedings of the 15th International Conference on*, pages 136–143, May-2 Jun 1995.

[2] Hari Balakrishnan, Srinivasan Seshan, Elan Amir, and Randy H. Katz. Improving TCP/IP performance over wireless networks. In *MobiCom '95: Proceedings of the 1st annual international conference on Mobile computing and networking*, pages 2–11, New York, NY, USA, 1995. ACM.

[3] J. Yu, S. Choi, , and J. Lee. Enhancement of VoIP over ieee 802.11 WLAN via dual queue strategy. In *ICC*, pages 3706–3711, 2004.

[4] Sherif M. ElRakabawy, Alexander Klemm, and Christoph Lindemann. Gateway adaptive pacing for TCP across multihop wireless networks and the internet. In *ACM MSWiM*, Torremolinos, Spain, 2006.

[5] Huan-Yun Wei, Shih-Chiang Tsao, and Ying-Dar Lin. On shaping TCP traffic at edge gateways. *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, 2:833–839 Vol.2, Nov.-3 Dec. 2004.

[6] Hong Fei and Bai Yu. Performance Evaluation of Wireless Mesh Networks with Self-Similar Traffic. *Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007. International Conference on*, pages 1697–1700, Sept. 2007.

[7] R. G. Cole and J. Rosenbluth. Voice over IP performance monitoring. In *Computer Communication Review*, volume 31, pages 9–24, April 2001.

[8] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla. The impact of multihop wireless channel on TCP throughput and loss. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 3:1744–1753 vol.3, March-3 April 2003.

[9] S. Xu and T. Saadawi. Does the IEEE 802.11 MAC protocol work well in multihop wireless ad hoc networks? *Communications Magazine, IEEE*, 39(6):130–137, Jun 2001.

[10] W. Stevens. TCP Slow Start, Congestion Avoidance, Fast Retransmit. *RFC 2001*, January 1997.

[11] L.S. Brakmo and L.L. Peterson. Tcp vegas: end to end congestion avoidance on a global internet. *Selected Areas in Communications, IEEE Journal on*, 13(8):1465–1480, Oct 1995.

[12] M. Gerla, M.Y. Sanadidi, Ren Wang, A. Zanella, C. Casetti, and S. Mascolo. Tcp westwood: congestion window control using bandwidth estimation. *Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE*, 3:1698–1702 vol.3, 2001.

[13] K. Tan, J. Song, Q. Zhang, and M. Sridharan. A Compound TCP Approach for High-Speed and Long Distance Networks. *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–12, April 2006.

[14] L. Xu and I. Rhee. CUBIC: A new TCP-Friendly high-speed TCP variant. In *Third International Workshop on Protocols for Fast Long-Distance Networks (PFLDNet06)*, Feb. 2005.

[15] Kai Chen, Yuan Xue, Samarth H. Shah, , and Klara Nahrstedt. Understanding bandwidth-delay product in mobile ad hoc networks. In *Special issue on protocol engineering for wired and wireless networks, Elsevier Computer Communications*, volume 27, pages 923–934, 2004.

Digital Object Identifier: 10.4108/ICST.WICON2010.8537
http://dx.doi.org/10.4108/ICST.WICON2010.8537