

AED: An Accurate and Explicit Loss Differentiation Mechanism

Vijay Arya — Thierry Turletti

N° 4587

October 2002

THÈME 1



*Rapport
de recherche*

AED: An Accurate and Explicit Loss Differentiation Mechanism

Vijay Arya , Thierry Turletti

Thème 1 — Réseaux et systèmes
Projet Planete

Rapport de recherche n° 4587 — October 2002 — 15 pages

Abstract: In recent years mobile computing has experienced an explosive growth, mainly due to the integration of wireless networks with the wired Internet. To deploy bandwidth-greedy multimedia applications on such heterogeneous environments, efficient congestion control algorithms are required. Indeed, in such a scenario, a packet can cross several wired and wireless links before eventually reaching its destination. When congestion control protocols such as TFRC are used to transmit multimedia flows, they need to differentiate between congestion losses and wireless losses to behave correctly and efficiently. Currently proposed end-to-end loss differentiation mechanisms can not reliably predict the differences between congestion and wireless losses. In this paper, we explore explicit loss differentiation. We have designed a simple window framework to explicitly and accurately differentiate these losses. By deploying agents at the boundaries of wireless links, we show that we can efficiently implement our window-based scheme with low overhead. We also point out current limitations of our scheme.

Key-words: Congestion control, loss differentiation, mobile computing, multimedia transmission, TFRC

This work has been supported by the french ministry of industry in the context of the national project RNRT-VIP.

AED: Un Mécanisme Précis et Explicite de Différenciation de Pertes de Paquets

Résumé :

Depuis quelques années, l'Internet mobile connaît une phase de croissance spectaculaire, principalement en raison de l'intégration des réseaux sans-fil aux réseaux filaires classiques. Pour que les nouvelles applications multimédia, plus gourmande en bande passante, puissent fonctionner dans ce type d'environnement, on a besoin de mécanismes de contrôle de congestion qui soient efficaces avec ce type d'environnement hétérogène. En effet, dans ce type de réseau hybride, les paquets IP peuvent traverser plusieurs liaisons filaires et sans fil avant de pouvoir atteindre leurs destinations. Lorsqu'un protocole de contrôle de congestion comme TFRC est utilisé pour transmettre des flots multimédia, il est important qu'il puisse distinguer les pertes de paquets causées par la congestion dans le réseau de celles provoquées par les erreurs de transmission sur les liaisons sans fil. Plusieurs schémas de différenciation de bout-en-bout ont été proposés mais ils ne permettent pas de faire la différence de manière fiable entre les deux types de pertes sur le réseau. Dans ce rapport, nous décrivons un mécanisme de différenciation explicite. Le mécanisme que nous avons élaboré utilise des agents placés à la frontière des réseaux sans fil. Nous montrons dans ce rapport qu'un tel mécanisme peut s'implanter à faible coût, et évaluons ses limitations.

Mots-clés : Contrôle de congestion, mécanisme de différenciation de pertes de paquets, réseaux IP hybrides, TFRC, transmission multimédia

1 Introduction

The Internet today is no longer completely wired, with Wireless LANs (WLANs), wireless backbones, and mobile networks getting appended to it. Most business offices, university campuses and hospitals too have WLANs. Both GSM(GPRS) and UMTS have standards for packet access from mobile terminals. Simultaneously, there is a growing popularity of real-time multimedia applications on the Internet such as audio/video streaming, IP telephony, video conferencing, network games, etc. The 3G wireless service providers are promising packet switched real-time multimedia services to their users and it is believed that a 3G terminal will be able to access and play streaming audio or video available on a server in the Internet. To support such real-time multimedia applications on a network with wired and wireless links, efficient congestion control mechanisms are required for real-time flows.

TCP, the most dominant protocol in the Internet, is not suited for transferring real-time flows. TCP guarantees reliability and packet ordering at the expense of increase in end-to-end delay. Reliability however, is not a stringent requirement for real-time data. Instead real-time data is timely and if it does not arrive after some deadline, it may not even be useful. Besides, TCP uses additive increase multiplicative decrease algorithm which shows high variation in sending rate resulting in large jitters, degrading the user perceived quality. For these reasons, real-time flows are carried either directly using UDP or using protocols such as RTP (over UDP) [13]. RTP does not specify any explicit congestion control mechanism like TCP. In order to be fair to protocols that use congestion control, such as TCP and to avoid any possible congestion collapse, several TCP-friendly congestion control mechanisms were proposed for unreliable unicast flows [6, 12, 3, 15]. But all these mechanisms, like TCP, rely on implicit feedback and interpret any loss as a congestion loss. The implicit policy works accurately in wired networks where there are almost no losses on wired links. However, with the presence of wireless links, a large percentage of packets are lost on these links. Wireless links are inherently error prone due to effects such as fading and multi-path and the wireless data link layer provides only partial reliability. Thus when TCP-friendly protocols are deployed on networks such as the Internet, they reduce their sending rate not only in response to congestion, but also in response to wireless losses. To avoid this false behavior, several end-to-end loss differentiation schemes were proposed [4, 14, 5]. These schemes however, misclassify wireless and congestion losses. If congestion losses are misclassified as wireless, the scheme may not be TCP-friendly and may cause more congestion in the network. If wireless losses are misclassified as congestion, the scheme may not use the available bandwidth. Thus, there is a need to accurately differentiate between wireless and congestion losses.

In this paper, we explore an explicit mechanism to accurately differentiate between wireless and congestion losses by deploying agents at the boundaries of wireless links. Our scheme is called *Accurate and Explicit Differentiation* (AED) scheme. We consider a real-time flow which uses a protocol such as RTP [13] or DCCP [9] in conjunction with TFRC [7]. AED aims to inform the TFRC receiver about wireless and congestion losses so that it can send an accurate feedback to the sender. The sender would use the congestion loss information in the feedback to adjust its sending rate based on the exact level of congestion in the network.

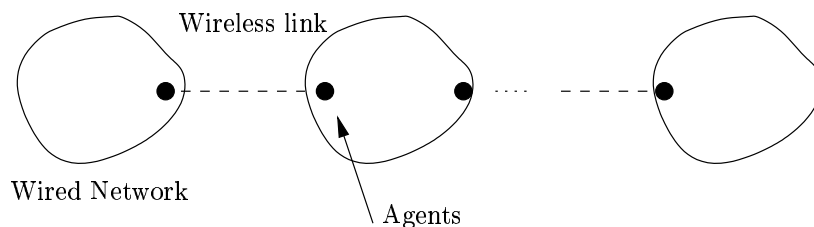


Figure 1: General topology of a hybrid network

It can also use the wireless loss information in the feedback to determine the amount of bits to be allocated for source and channel coding, so that more losses can be recovered without retransmissions. Balance of the paper is organized as follows. Section 2 discusses the past work, Section 3 explains our scheme, sections 4 and 5 give the implementation details and limitations respectively. We conclude with section 6.

2 Related work

The aim of loss differentiation schemes is to differentiate between wireless and congestion losses. These mechanisms are either *End-to-end* or *Explicit*. Explicit schemes are those that make use of agents deployed on intermediate network nodes. End-to-end schemes try to differentiate losses at the receiver without making use of any intermediate nodes.

Explicit Loss Differentiation was made use of in the context of TCP flows by the use of *Snoop Agents* [2]. Snoop agents are suited for first hop/last hop wireless topology encountered in mobile networks. Consider a TCP flow originating from a Fixed Host (FH) and terminating at a Mobile Host (MH). The snoop agent at the base station, by monitoring the TCP packets forwarded to MH and acknowledgments returned from the MH, maintains a cache of TCP packets that have been forwarded but not acknowledged by the MH. It detects a loss of packet on the wireless link by seeing duplicate acks or by a timeout based on locally maintained timers. When it does, it retransmits the lost packet to the MH if it has been cached, since these are packets lost on the wireless link. Additionally, it suppresses duplicate acks corresponding to the wireless losses, thus ensuring that the false congestion invocations at the sender are avoided. Note firstly that snoop agent decouples wireless from congestion losses without making any changes to the IP packets itself. Secondly, snoop works for reliable protocols since it also retransmits the lost packets cached.

Snoop agents are not suited for real-time flows since these flows are carried by unreliable protocols. For snoops to detect packet loss, the receiver needs to acknowledge every packet received. Although rate based protocols are more suited for real-time flows, snoop agents may be a possible candidate for congestion control mechanisms like Binomial congestion control [3] (where the receiver acknowledges every packet like TCP). But suppressing the duplicate acknowledgments for packets lost on the wireless link (to decouple wireless and

congestion losses), may increase the end-to-end delay which is harmful for real-time applications. For real-time applications, local retransmissions also may not help as retransmission delay may sometimes be unbearable. For real-time flows, agents which send information either to sender or receiver about wireless and congestion losses, by either marking packets or generating new packets, are required.

ECN (Explicit Congestion Notification) [8] is also a possible candidate for real-time flows. ECN is used by the routers to signal congestion in the network by marking IP packets. When TFRC is used in conjunction with ECN, the losses detected when marked IP packets are received are considered congestion losses and those detected when unmarked IP packets are received, are considered to be wireless losses and are not used to update loss probability. But ECN can be inaccurate. For example, for a continuous bunch of wireless and congestion losses, ECN would classify all of them to be congestion losses. Although it may be the right time to invoke congestion control by reducing the sending rate, the amount of reduction will be more than necessary.

End-to-end schemes work at the transport level in the receiver. They make use of facts such as the time taken by the packet to travel through the network and packet inter-arrival time. Vidya and Biaz [4] proposed a scheme based on packet inter-arrival time suitable for last hop wireless bottleneck link. They approximate the minimum packet inter-arrival time seen during a connection (T_{min}) to the time taken by a packet to traverse the wireless link. If a packet is lost and the subsequent packet arrives approximately after T_{min} , it is classified as a wireless loss. This scheme has two main problems. Firstly, with more than one flow, packets from different flows get interspersed and packet after a wireless loss may not come immediately after T_{min} . Secondly, in practice, the wireless channel is generally varying and the time taken by a packet to traverse the wireless link keeps changing, causing misclassification. Tobe et al, [14] use Relative One-way Trip Time (ROTT) as a measure of the time taken by a packet to travel from source to destination. Their scheme is based on the fact that when ROTT is plotted against time at the receiver, one can observe spikes during congestion. Thus losses during spikes are classified as congestion losses, and otherwise wireless. The problem here is that congestion can occur on one or more routers together, resulting in spikes of varying sizes. The difficult part is to catch spikes of all sizes. The Zig-zag scheme [5] also makes use of ROTT. In this scheme, the receiver keeps an estimate of running $ROTT_{mean}$ and $ROTT_{variance}$. Based on the number of losses and the difference of the current ROTT and $ROTT_{mean}$, they propose certain conditions for classification. All the end-to-end schemes were compared in [5]. End-to-end schemes generally have topological constraints. Although, some of these do improve TFRC throughput, they all suffer from significant misclassification. Thus congestion losses in the network may increase and TCP-friendliness of the flow may decrease. Also, the sender is unable to use receiver's feedback for adjusting the bits allocated for source and channel coding since the feedback is inaccurate.

We consider an explicit loss differentiation mechanism which is both accurate and is not limited by any topological constraints. We consider a general topology and show how by deploying agents at the boundaries of wireless links, we can accurately differentiate the losses. Using our AED scheme, TFRC is able to function like the optimal Omniscient-TFRC

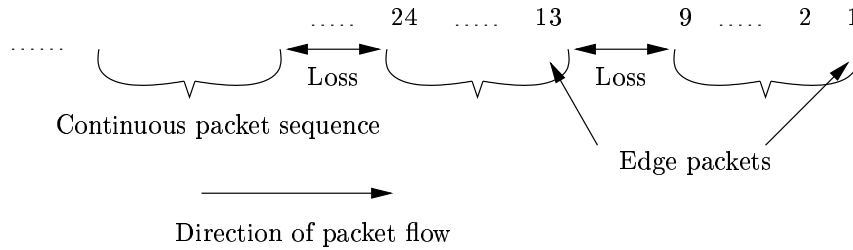


Figure 2: Packet sequence as seen by an agent

used for comparison in [5]. Omniscient-TFRC knows precisely which losses are congestion and which wireless and thus performs optimally on all topologies. In subsequent sections, we give the details of our scheme.

3 The AED Mechanism

We consider the following general topology of a hybrid network (figure 1) with a series of wired subnetworks interleaved by wireless links. Each subnetwork consists of at least one node. For example, a mobile node is considered as a wired network with one node.

Packets are either lost due to congestion on the wired subnetworks or on the wireless links. To determine the right cause of packet loss, we deploy agents before and after each wireless link. The agents snoop through each packet and detect a loss by seeing a packet with an out of order sequence number. For the packets flowing from left to right in Figure 1, agents at the end of wired networks treat a packet loss as congestion loss and agents after a wireless link treat a packet loss as wireless loss.

Figure 2 shows a general packet sequence with interleaved losses, as seen by an agent. The agent would want to mark the packets which are not lost, with information about the lost packets, that is, whether packets were lost due to congestion or a wireless link. However, the information must be recorded using a scheme which is most immune to future packet losses, so that the information finally reaches the receiver. For example, consider a scheme in which the agent marks the edge packets (packets just after a loss). Suppose that in Figure 2, packets 10..12 were lost on a wireless link and edge packet 13 holds this information. If packet 13 itself is lost by congestion in future, we would lose all the information as to how 10..12 were lost.

We consider a generalized scheme in which each packet holds information about a window of previous n packets. We call such a window as the *history* window. For a packet with sequence number j , its history window holds information about packets $j - 1, \dots, j - n$, as to whether these packets were lost due to congestion, wireless link, or not lost at all. Figure 3 shows the concept of keeping a history window of size four. In this example, packet 5 is considered to be lost by congestion and 4 on a wireless link. The history windows maintained by packets 6 and 7 have been shown. 6's window records the fact that 5 is a congestion loss,

4 a wireless loss and that 3 and 2 are not lost. With this information, if packet 6 is lost in future due to congestion or a wireless link, we still will not lose information as to how packets 5 and 4 were lost since this information would also be available in packet 7. Only if all packets 6, 7, 8 were lost by congestion, we would lose the information as to how 4 was lost.

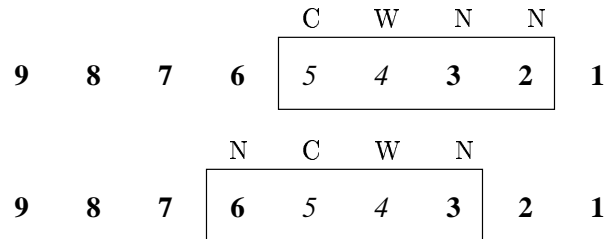


Figure 3: *History* window concept

Our next step is to determine how to record the contents of the history window efficiently in each packet. We propose a scheme which can represent a history window of size n , using just n bits. Each history window may consist of wireless losses, congestion losses and no

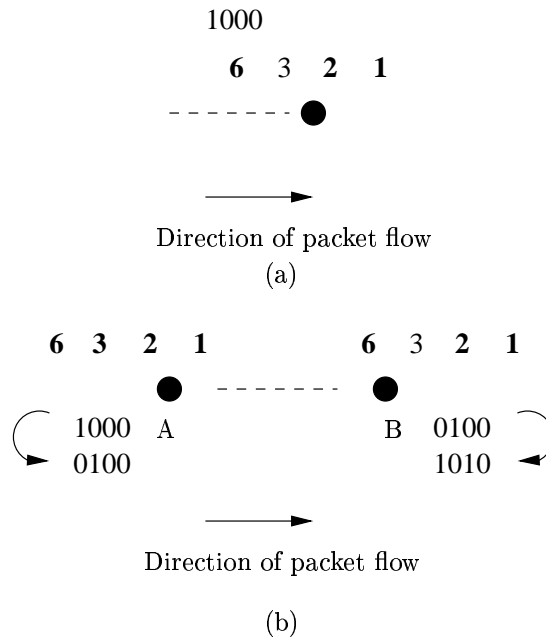
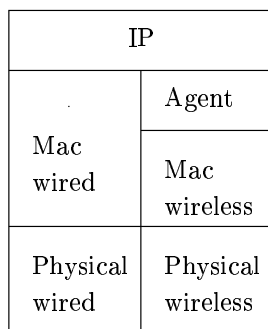


Figure 4: Flipping the bits before and after the wireless link

losses. Now, suppose that we represent a wireless loss by 1 and congestion loss or no loss by 0, then we would face the following problem. Consider the scenario shown in Figure 4(a). In this example, 5 was lost on a wireless link somewhere before, 4 was lost due to congestion in the previous wired subnetwork. Hence 6's window is represented as 1000. Suppose that 3 gets lost on the wireless link. Now when the agent sees the packet sequence, it finds 5,4,3 missing. Since 5 is represented by a 1 in 6's window, it knows 5 was lost by a wireless loss. But it is unable to differentiate packets 4 and 3 in spite of the fact that they were lost because of different reasons. This problem can be tackled without increasing the number of bits to represent the history window, if each agent flips the bits appropriately. The point to observe is that the agent after a wireless link tries to detect *wireless* losses by seeing missing sequence numbers. So we must tell it explicitly if certain missing sequence numbers were due to congestion earlier. Hence this agent expects to see congestion losses represented by a 1. The reverse is true for the agent before the wireless links which aims to determine *congestion* losses by seeing missing sequence numbers. We must tell it explicitly if certain missing sequence numbers were due to a wireless link earlier.

Thus the agent before a wireless link interprets a wireless loss by 1 and congestion or no loss by 0. But before sending the packets on the wireless link, it flips the bits such that congestion loss is represented by 1 and wireless loss or no loss is represented by 0. The agent after the wireless link interprets congestion losses by 1 and wireless loss or no loss by a 0. But before sending the packets on the wired network, it flips the bits again such that wireless loss is represented by 1 and congestion or no loss is represented by 0. For the example in Figure 4(b), agent A doesn't see packets 5,4. It knows that 5 was lost by wireless loss since it is represented by 1 in 6's window. It concludes that 4 was lost by congestion, hence it flips the bits and represents 6's window by 0100. Now 3 gets lost on the wireless link. Agent B doesn't see 5, 4 and 3. Seeing 4 represented by 1, it concludes that 4 was a congestion loss and treats 5 and 3 as wireless losses. Hence it flips the bits again to 1010. In this manner, the receiver will obtain wireless losses represented by a 1 and congestion losses represented by a 0. Note that if the receiver is a mobile node, it would have an agent inside it too.

Observe that packet reordering does not affect this scheme. Each packet will always hold the information about packets with preceding n sequence numbers, which are seen (or seen missing) ahead of it by the agent. The receiver will use the window information about a packet only if it doesn't receive that packet. For example, suppose that an agent sees packets 1..5 and then sees 7 followed by 6. 7's window will note 6 as lost and 6's window will note information starting from 5. To accommodate for packet reordering, the receiver can wait for certain number of subsequent packets before concluding a loss (as 3 packets in TFRC and TCP). It is important to observe two points here. Firstly, the agent simply inspects and labels a packet based on previously seen packets. It does not hold any packets either before or after the current packet. Secondly, the wireless and congestion losses will never be misclassified within the window. Although, it is possible that a packet which is not actually lost is marked lost within the window (due to reordering) or the information about a loss is not known (if the window is not large enough).



(a)



Header Payload

(b)

Figure 5: Packet structure and Agent implementation

4 Implementation of AED

4.1 AED Agent Implementation

The AED agent can be deployed in between the wireless data link layer and the IP layer as shown in Figure 5. We show here, the implementation of the AED agent with DCCP [9]. Figure 5 also shows the packet structure which the agent sees. Each DCCP packet is shown encapsulated in a single IP packet (TFRC details have not been shown here). The agent uses the DCCP sequence number to detect packet loss. In the previous section, while explaining the mechanism, we considered packets flowing in one direction. In practice, both the agents will function in the same dual manner. When the IP packet is passed from the wireless data link layer to the IP layer, wireless losses will be detected and the agent flips the bits such that wireless loss is represented by a 1 and congestion or no loss by a 0 (agent after wireless link). When the packet is passed from IP layer to the wireless data link layer, congestion losses will be detected and the agent flips the bits such that congestion loss is represented by a 1 and wireless or no loss by a 0 (agent before wireless link).

The agent distinguishes between flows by looking at the tuple (source IP, dest IP, source port, dest port) in each IP packet. The TOS field in the IP packet can be used to detect if a flow is a DCCP flow. Since DCCP supports connection setup and tear-down, the agent knows when to allocate and free memory for a flow. An additional bit will be required to determine if a DCCP flow supports the use of AED or some other loss differentiation

mechanism. The agent uses a space of n bits in the DCCP header to record the history window. For each flow, each agent maintains a history of previous $n + k$ sequence numbers seen, where k serves as cushion for packet reordering. k could be set 3 as in TCP and TFRC. Thus per-flow memory overhead for the agent is only $O(n + k)$. Note here that, with this implementation, the losses occurring within the IP layer will be classified as congestion losses. Any losses occurring below the wireless MAC layer will be classified as wireless (including possible losses at buffers within the wireless Physical/MAC layer).

4.2 Receiver Implementation

The transport level implementation at the receiver is minimal. When the receiver receives an out of order DCCP packet, it will simply need to read the n window bits of this packet, to conclude the reasons of loss for the missing packets. With the convention used in section 3, if the window bit is set 1, it is a wireless loss and if it is set 0, it is a congestion loss. To deal with packet reordering the receiver will have to wait for k additional packets to find out if a packet is actually lost. In case of TFRC, the congestion losses will be used for calculation of loss rate.

4.3 Window size

To implement the AED scheme, it is important to decide what window size to use. In this section, we try to find this using simulation. A large window size would cover all losses, but it would also increase the overhead on the amount of data to be transmitted. A small window may miss some losses but requires less extra bits to be transmitted. There is a tradeoff between amount of losses covered and the corresponding additional overhead. The optimal window size depends on the distribution of loss burst lengths. The loss bursts due to congestion are not of much consequence since the flow responds to congestion by reducing the sending rate. It is the wireless loss bursts which will have an impact on the window size.

[11] and [1] studied UDP packet traces on real wireless links to obtain a model for wireless channel loss behavior. (They conducted their experiments using WaveLAN links in REINAS and BARWAN networks respectively). They observed that for 90% of loss bursts, the burst length was less than or equal to 4 packets. [11] obtained Simple and Improved two-state markov models using the trace data that they collected. For their simple two-state markov model (STMM) shown in Figure 6(a), $P_{GE} = 1/L_G$ and $P_{EG} = 1/L_E$. L_E and L_G are average error and error-free burst lengths respectively in their UDP traces. Based on the trace data, [11] set $L_E = 2.61770277$ and $L_G = 166.394284$.

Next, we consider a simple topology with two wireless links separated by a wired sub-network shown in Figure 6(b). For each wireless link, we simulate losses using STMM and we simulate congestion losses in the wired subnetwork using a uniform random packet loss probability of 1%. We do this for 10^6 packets and check the loss burst lengths seen by the receiver. Figure 7 shows the histogram of these loss burst lengths and Table 1 shows the corresponding loss statistics. A window of size W covers all loss bursts of length less than or equal to W and has an overhead of W bits per packet. Table 2 shows the losses

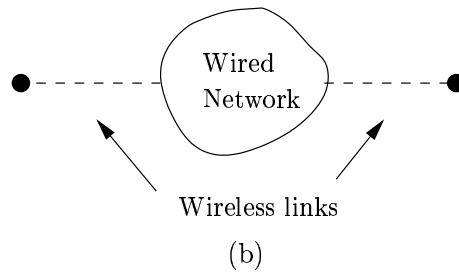
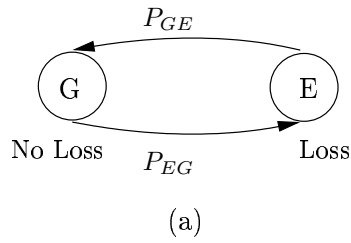


Figure 6: STMM and Topology used for simulation

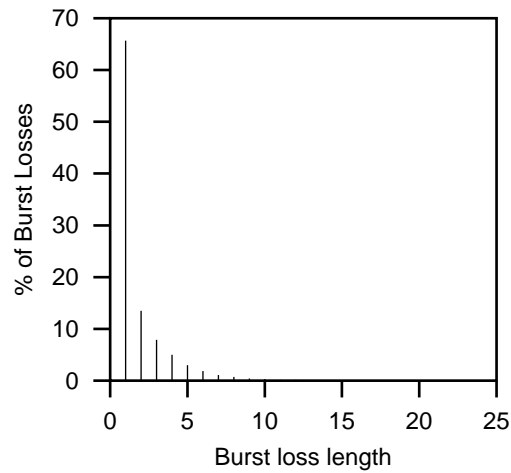


Figure 7: Distribution of burst loss lengths

covered by windows of various sizes versus the extra overhead for the connection. For bursts larger than the window size, the cause of certain losses is not known. If these unknown losses as classified as congestion, certain wireless losses will get misclassified as congestion losses. Table 3 shows these misclassification rates. Based on the notation in [5], $M_w =$

Total packets	10^6
No.of loss bursts	20879
Maximum burst length	22 (1 burst)
Gross loss rate	3.96%
Congestion losses	25.14 % of losses
Wireless losses	74.86 % of losses

Table 1: Loss statistics

Window (W)	Bursts $\leq W$	Losses covered by W	Total Overhead(KB)
4	92.03 %	71.85 %	500
8	98.72 %	92.77 %	1000
16	99.98 %	99.75 %	2000

Table 2: Losses covered by windows of different sizes

W	Wireless losses misclassified (% of total losses)	M_w	M_c	False congestion rate	TFRC throughput(%)
4	11.24 %	15.02 %	0	1.4452 %	83.19 %
8	1.83 %	2.44 %	0	1.0725 %	96.56 %
16	0.045 %	0.06 %	0	1.0018 %	99.91 %

Table 3: Misclassification if unknown losses are considered congestion losses

(No. of wireless losses misclassified as congestion)/(total wireless losses) and $M_c =$ (No. of congestion losses misclassified as wireless)/(total congestion losses). From [6], we know that TFRC throughput is proportional to $S/R\sqrt{p}$, with usual notation. We use this to find out the impact of window size on TFRC throughput. For our simulation, the congestion loss rate p was 1%. Thus omniscient-TFRC would have throughput proportional to $S/R\sqrt{0.01}$. But misclassifying wireless losses as congestion would result in a wrong increased loss rate (p_w), the receiver calculates, reducing TFRC throughput. In the last two columns of Table 3, we have calculated the false congestion loss rate and TFRC throughput ratio (w.r.t omniscient-TFRC i.e., $\sqrt{p/p_w}$), for various window sizes (neglecting the increase in packet size). We can see from Table 2 and 3 that a window size of 8 provides reasonably good throughput.

5 Limitations

5.1 Header compression and IP security

In case of header compression at the IP layer, with IP and DCCP headers compressed, the AED agent will not be able to inspect the DCCP sequence number to detect loss. In order to do so, either it needs to be implemented within the IP layer or it remains outside and performs another independent compression and decompression. When protocols like IPSec are used, in case of authentication, the agent can see but cannot change the DCCP header and when the IP packet is encrypted, it is not possible to both see and change the DCCP header present in the IP payload.

5.2 Increase of RTT

Although the procedure of inspecting and labeling the packets is extremely simple and can be implemented efficiently, if a packet passes through several agents it may result in slight increase of RTT.

5.3 Fragmentation

We assumed in the AED scheme that an entire DCCP packet is either lost or not lost. In case there is fragmentation of IP packets in the network, the loss of a DCCP packet will be detected and recorded only if IP packet containing the DCCP header(sequence number) would be lost. There is also a question of how to classify the loss during fragmentation. Suppose that an IP packet containing a DCCP packet is fragmented into two IP packets, one of which is lost during congestion and the other on a wireless link. But fragmentation is not such a serious problem. [10] made an empirical study of Internet traffic. In the set of 60 traffic traces from the Internet that they collected, majority of them had fragmentation levels below 1%. Hence we would have very few or no packets per flow for whom we do not know the precise cause of loss.

5.4 Huge packet gaps

We believe the topology shown in Figure 1 is the general topology of hybrid networks, but one cannot completely rule out cases in which a single flow splits and passes through two different wireless links. This is one of the cases when the agents will see huge packet gaps. But the packets in the gaps are not actually lost and can reach the receiver. It would be more useful to store information about packets ahead of the gap since these can get lost. When packet gaps exceed the window size, our scheme of defining window, will not be able to record information about packets ahead of the gap. However, other schemes of defining and recording windows will add more overhead in terms of number of additional bits to be transmitted.

6 Conclusions

Accurate differentiation of wireless and congestion losses is necessary for proper functioning of several transport protocols on hybrid networks. In this paper, we have explored the design and implementation requirements of an efficient and accurate explicit loss differentiation mechanism. We saw that by deploying agents which are aware of higher level protocols, one can accurately distinguish wireless from congestion losses. The success of several hybrid 3G networks depends on the performance of real-time protocols on these networks. We believe that explicit loss differentiation is a possibility in networks such as UMTS where agents can be deployed in gateway nodes such as GGSN. We also studied the possible window sizes to use with AED, using simulation. In future works, we will investigate theoretically, how window size affects the TFRC throughput.

Acknowledgements

We would like to thank Chadi Barakat for useful feedback and discussions.

References

- [1] Hari Balakrishnan and Randy Katz. Explicit loss notification and wireless web performance. In *IEEE GLOBECOM Internet Mini-Conference.*, Sydney, Australia, November 1998.
- [2] Hari Balakrishnan, Srinivasan Seshan, and Randy H. Katz. Improving reliable transport and handoff performance in cellular wireless networks. *ACM Wireless Networks*, December 1995.
- [3] Deepak Bansal and Hari Balakrishnan. Binomial congestion control algorithms. In *IEEE INFOCOM*, pages 631–640, April 2001.
- [4] S. Biaz and N.H. Vaidya. Discriminating congestion losses from wireless losses using inter-arrival times at the receiver. *IEEE Symposium ASSET'99, Richardson, TX, USA*, March 1999.
- [5] S. Cen, P.C. Cosman, and G.M. Voelker. End-to-end differentiation of congestion and wireless losses. *Multimedia Computing and Networking (MMCN) conf. 2002*, pages 1–15, January 2002.
- [6] Sally Floyd, Mark Handley, Jitendra Padhye, and Jorg Widmer. Equation-based congestion control for unicast applications. In *SIGCOMM*, pages 43–56, Stockholm, Sweden, August 2000.
- [7] M. Handley, J. Padhye, and S. Floyd. TCP Friendly Rate Control(TFRC): Protocol specification, April 2002. <http://www.icir.org/tfrc/>.

-
- [8] Ramakrishnan. K.K, S. Floyd, and D. Black. Addition of ECN to IP. RFC 3168, September 2001. <http://www.icir.org/floyd/ecn.html>.
 - [9] E. Kohler, M. Handley, J. Padhye, and S. Floyd. Datagram congestion control protocol(DCCP), May 2002. <http://www.icir.org/kohler/dcp/>.
 - [10] F. Li, N. Seddigh, B. Nandy, and D. Malute. An empirical study of today's internet traffic for differentiated services IP QoS. In *Proc. of ISCC*, 2000.
 - [11] Giao Thanh Nguyen, Randy H. Katz, Brian Noble, and M. Satyanarayanan. A trace-based approach for modeling wireless channel behavior. In *Winter Simulation Conference*, pages 597–604, 1996.
 - [12] I. Rhee, V. Ozdemir, and Y. Li. TEAR: TCP emulation at receivers-flow control for multimedia streaming. Technical report, Department of Computer Science, NCSU, April 2000.
 - [13] H. Schulzrine, S. Casner, R. Frederick, and V. Jacobson. RTP: A transport protocol for real-time applications. RFC 1889, January 1996.
 - [14] Y. Tobe, Y. Tamura, A. Molano, S. Ghosh, and H. Tokuda. Acheiving moderate fairness for UDP flows by pathstatus classification. In *25th Annual IEEE Conf. on Local Computer Networks*, pages 252–61, November 2000.
 - [15] Y. R. Yang and S. S. Lam. General AIMD congestion control. In *Proceedings of 8th ICNP*, November 2000.



Unité de recherche INRIA Sophia Antipolis

2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)

Unité de recherche INRIA Rennes : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)

Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38330 Montbonnot-St-Martin (France)

Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)

Éditeur

INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)

<http://www.inria.fr>

ISSN 0249-6399