CHAPTER 5
DIFFERENTIATION OF CONGESTION AND RANDOM LOSSES

5.0 INTRODUCTION

In the current scenario of Internet environment, the logical connections are made up of both wired and wireless. The wireless networks are always prone to errors. Hence the losses in the wireless network could also be detected as congestion losses. The key idea of this chapter is to eliminate the ambiguities arising out of the random losses viewed as congestion losses. This chapter presents a biased queue management scheme at the router. This enables the TCP receiver to distinguish between the congestion losses and random losses and inform the TCP sender to react appropriately. Bounds on the accuracy of distinguishing wireless losses and congestion losses are analyzed. Congestion losses are identified with accuracy higher than 95%.

Access to Internet through wireless media has become very common. Currently, a TCP sender considers all losses as an indication of congestion and reacts to them by throttling its sending rate. But the loss may be due to congestion control at the router or the random losses in the wireless environment. Owing to the heterogeneity of the components in the Internet, the causes of packet losses are not predictable. These losses are misinterpreted as congestion losses by TCP sender, which throttles down the rate of the data degrading the performance of the network. But the problem of distinguishing congestion losses from random losses is particularly hard when congestion is light. Despite efficient link layer techniques [using Automatic Repeat reQuest (ARQ) and/or Forward Error Correction (FEC)], some losses unrelated to congestion may still occur. These errors have to be ultimately dealt with transport
layer. TCP is the most popular transport protocol used over the Internet carrying more than 90% (C. Fraleigh, S. Moon, C. Diot, B. Lyles, and F. Tobagi, 2001) of all Internet traffic. The problem of TCP over links prone to random losses is well known (H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz, 1996), since random transmission losses unnecessarily trigger TCP's congestion control mechanisms. Though there are good numbers of end-to-end discriminators are available (S. Biaz and N. H. Vaidya, 1998), (S. Cen, P. C. Cosman, and G. M. Voelker, 2003), (T. Kim, S. Lu, and V. Bhargavan, 1999), (J. Liu, I. Matta, and M. Crovella, 2003), (C. Paras and J. J. Garcia-Luna-Aceves, 2000), (N. Samaraweera, 1999), their performance is not satisfactory enough to be widely deployed. To solve this problem, a promising approach is to eliminate the ambiguities in the type of losses sensed by TCP sender. With such an approach, when losses occur, it can be resolved by using the answer for a question like “Is this kind of loss random?” If the answer is ‘no’, then the losses could be interpreted as congestion losses; otherwise it is random losses. Hence a discriminator for this purpose is built upon two components:

1. The segments being sent are added with different discard priorities
2. A Biased Queue Management (BQM) technique can be used in the router to eliminate the ambiguities by implementing the discard policies.

By having different dropping priorities, the probability distribution of congestion losses will be different for the two types of packets. The key idea of proposed method is to send, within the same TCP connection, packets with different markings for the router to decide for dropping.
A service similar to RIO (RED In/Out) (D. D. Clark and W. Fang, 1998) is expected from the network provider. In RIO packets are marked “in” or “out” by an edge router such that core routers preferentially drop packets marked out in case of congestion. The major differences between RIO and the proposed scheme:

1) Packets are marked “in” or “out” by the TCP sender (endpoint);
2) Packets of the same TCP connection get different discard priorities; and
3) Packets marked out are dropped first before any packet marked in is.

With RIO, 1) packets are tagged by edge routers; 2) packets of the same TCP connection are marked the same; and 3) a packet marked in may well get dropped while a packet marked “out” is not. Only the router interprets the markings such as in and out. By these different discard priorities it is possible to distinguish congestion losses from random losses.

This chapter focuses on the end-to-end solutions to enhance TCP over networks where random losses may be unrelated to congestion. There are mainly two approaches toward end-to-end solutions for enhancing TCP dealing with losses unrelated to congestion. The first approach consists in distinguishing random losses from congestion losses and makes TCP react appropriately to each kind of loss. The second approach does not use losses as a congestion measure to control the sending rate. Instead, it attempts to estimate the available bottleneck link capacity and adopts an adequate sending rate that optimally exploits this available capacity. This approach does not require the discrimination of losses because losses are not used to control the sending rate. For the first approach, Biaz and Vaidya (S. Biaz and N. H. Vaidya, 1998) studied the ability of three loss discriminators based on congestion predictors: Jain’s delay-based congestion
predictor \textit{(D. Chiu and R. Jain, 1989)}, Wang's throughput-based predictor \textit{(Z. Wang and J. Crowcroft, 1991)}, and Vegas predictor \textit{(L. S. Brakmo and S. O'Malley, 1994)}. In this chapter, a simple biased queue management scheme is proposed as the active queue management (AQM) technique to distinguish congestion losses from random wireless losses. When TCP traffic is predominant (higher than 85%), simulation results show that the proposed discriminator consistently achieves high accuracy (more than 95%) in identifying congestion losses and high accuracy in identifying wireless losses (more than 75%).

The rest of the chapter is organized as follows. Section 5.1 discusses the basics of the discriminator. The appropriate action that TCP should take when random losses unrelated to congestion are detected is discussed in Section 5.2. Section 5.3 describes the results of experiments. Section 5.4 concludes the chapter.

5.1 Basics of Proposed Discriminator

The proposed discriminator is implemented at the TCP receiver because the receiver has a better view of the losses than sender.

\textbf{Fig.5.1 Schematic block diagram of Discriminator connection}
Consider Fig. 5.1 that illustrates a pattern of losses that would likely result from a biased drop as opposed from a wireless random dropping. Fig. 5.1 presents a sender and a receiver with a path that includes a bottleneck link implementing a biased queue management and a wireless link. The sender marks some of the packets as ‘out’ and the remaining packets marked as ‘in’. The number of packets marked ‘out’ is one out of $k$ packets. In case of congestion, the router at the bottleneck link must first drop packets marked “out” before dropping any packet marked “in”. If there is no congestion, the pattern of dropping will likely be different because, as shown in Fig. 5.1 bottom portion, the wireless link does not distinguish between $out$ and $in$ packets. Since there are a high proportion of packets marked “in”, it is expected that these will likely be dropped. From this observation, the receiver will diagnose a pattern of losses as biased if a high proportion of packets marked $out$ are lost. If the pattern appears to be biased, the receiver concludes that the losses are due to congestion.

Let us denote all data packets a TCP sender sends as $p_i$ where it is the $i$th packet, and it is marked with $M$ (in or out). A retransmitted packet keeps the same index but is always marked $in$ to avoid double jeopardy for packets initially marked $out$. It is assumed that routers first drop packets marked $out$ and start dropping packets marked $in$ only if there are no more packets marked $out$ in the queue. TCP sender marks the packets such that one packet out of $k$ ($k \geq 2$) packets is marked $out$ (all other packets are marked $in$).

Whenever an out-of-order packet is received, a loss is assumed to have occurred. TCP receiver considers the pattern of losses between the next expected packet and the packet with the highest sequence number seen so far.
Let $S = (hi - nxt + 1)$ be the number of packets in the ordered set $\{P_{nxt}, \ldots, P_{hi}\}$. Suppose there are $r$ losses among the $S$ packets. The \textit{phenomenon of packet loss} may be considered as a random sampling of $r$ elements among a population of $S$ elements, which contains ‘n’ defective elements (here, a \textit{defective} element means a packet marked \textit{out}).

The probability of drawing (loosing) $x$ packets marked \textit{out} in a random sample of size $r$ among a population of $S$ packets follows a hyper geometric distribution. Let $X$ be a random variable representing the number of packets marked \textit{out} in a sample of $T$ packets taken among $S$ packets ($T$ packets among $S$ are lost). Denote $y = \lfloor (S/k) \rfloor$ the number of packets marked \textit{out} within $S$ packets is:

$$P(X = x) = \frac{\binom{y}{x} \binom{S - y}{r - x}}{\binom{S}{r}} \quad (5.1)$$

If the pattern of $r$ losses appears to be a result of a \textit{random} sampling, then the losses are considered as wireless. \textit{Randomness} of sampling is tested as follows: if a loss pattern has a very low probability to occur, then the sampling is not random, but rather biased. In case of doubt, it is safe to conclude that losses are due to congestion. From these observations, a very simple function is developed to summarize the pattern of the loss:

$$F(X, r, k) = 1 - \lfloor k \cdot \frac{x}{r} \rfloor \quad (5.2)$$

where $x$ is the number of lost packets marked \textit{out} and $r$ is the number of losses within the ordered set $\{P_{nxt}, \ldots, P_{hi}\}$. The function $F(x, r, k)$ is built upon the following justification:

if losses are wireless (random), it is expected to have $(x/r) \approx (1/k)$, if the losses are
uniformly distributed, and will be equal to zero. Recall that \((x/r)\) represents the proportion of lost packets marked \textit{out} (i.e. \(x\)) over the total number of losses (i.e., \(r\)). If the losses are due to congestion, it is expected that the proportion \((x/r)\) of packets marked \textit{out} be higher than \((1/k)\), making \(F(x, r, k)\) is negative. Smaller is \(F(x, r, k)\), higher is the likelihood of congestion losses. Note that \(1-k \leq F(x, r, k) \leq 1\). The function \(F(x, r, k)\) does not capture all the information that could be extracted from the pattern of losses. However, the function \(F(x, r, k)\) yields a simple, robust, proposed discriminator. Whenever \(r\) losses occur with \(x\) packets marked \textit{out}, they are diagnosed as follows: if \(F(x, r, k) \geq 0\), then the losses are diagnosed as wireless losses, otherwise (i.e. \(F(x, r, k) < 0\)) they are diagnosed as congestion losses.

Algorithm 1 and Algorithm 2 present how the proposed scheme is implemented at the server (sender) and client (receiver) sides. The TCP receiver makes the diagnosis of a loss when it generates a duplicate ack. As the TCP receiver receives further out of order packets, it keeps diagnosing losses and sending back its diagnosis with generated dupacks. If a wireless loss is diagnosed, the duplicate ack is marked with “ELN” (Explicit Loss Notification). The sender uses the marking on the third dupack to decide whether to halve or not the congestion window.

\textbf{Algorithm 1}

\begin{verbatim}
//TCP sender (Server)
For each acknowledgement packet
  If Dupack then
    If ELN bit is set then
      Do not reduce congestion window size
    End
  End
Normal TCP congestion control operation
End
\end{verbatim}

\textbf{Algorithm 2}

\begin{verbatim}
// Receiver (Client)
For each packet received
  If out of order packet then
    Compute F(x,r,k)
    If (F(x,r,k) >= 0) then
      Mark Ack with ELN
    End
  End
Normal TCP operation
End
\end{verbatim}
To measure the accuracy of the proposed method, two metrics are used: the congestion accuracy $A_c$ and the wireless accuracy $A_w$. $A_c$ is the ratio of the number of congestion losses correctly diagnosed over the total number of congestion losses. $A_w$ is similarly defined for wireless losses. The objective is to achieve congestion accuracy $A_c \approx 1$ because multiple congestion losses mistaken for wireless losses may significantly degrade the network performance, since TCP senders would not decrease their sending rate when they should. On the other hand, a lower accuracy $A_w$ for diagnosing wireless losses is acceptable since a wireless loss mistaken for congestion will only degrade the performance of this TCP connection that unduly throttles its sending rate. As of today, TCP has an accuracy $A_c$ of 1 (all losses are considered congestion losses) and an accuracy $A_w$ of 0. Our objective is to reach $A_c \geq 0.95$ and $A_w \geq 0.75$. The value of $k$ is critical to the accuracy of discriminator. Relationships between $k$, the accuracy $A_w$, the accuracy $A_c$, and the congestion packet loss rate $p_c$ are developed in the following. Also the upper bounds on $A_c$, $A_w$ and $k$ could be established. The discriminator is shown to be fully efficient only if $k \leq (1/p_c)$. Moreover, the impact of the bottleneck buffer size on $A_c$ is discussed.

**Upperbound on $A_w$:** Consider the set $W$ of wireless losses: the expected fraction of packets marked out in $W$ is $(1/k)$ because: (1) Wireless losses are assumed random, (2) the fraction of packets marked out is $(1/k)$, and (3) wireless loss phenomenon does not discriminate between packets marked in and out. If losses do not occur in batches then each wireless loss of a packet marked out will be incorrectly diagnosed as a congestion loss. In summary of all wireless losses, a fraction $(1/k)$ will be incorrectly diagnosed. So, the accuracy $A_w$ cannot be higher than $1- (1/k)$. Then
\[ A_w \leq 1 - \frac{1}{k} \]  

(5.3)

This bound assumes that a proportion of \((1/k)\) packets marked \textit{out} reach the wireless link: this assumption is only true when there is no congestion, or congestion is very low. Such a bound suggests taking a large value for \(k\). However, \(k\) cannot be large without limit.

\textbf{Upper bound on } \(A_c\): Consider a flow with a total number of 100 packets with one packet out of 10 marked \textit{out} (i.e. \(k = 10\)). A favorable case is when the congestion loss rate is less than 10\% (i.e., \(p_c \leq 1/k\)). In such a favorable case, less than 10 packets on average will be dropped. Since there are 10 packets marked \textit{out}, it is highly probable that BQM will find in the queue packets marked \textit{out} to drop and no packet marked \textit{in} would be dropped. Therefore, congestion losses will be correctly diagnosed. Under such favorable conditions, \(A_c\) can reach 1.

Now, suppose that the congestion loss rate is higher than 10\% (i.e., \(p_c \geq (1/k)\)). Let \(p_c\) be for example 15\%. Then 15 packets will be dropped. Since there are only 10 packets marked \textit{out}, then 5 packets marked \textit{in} will be dropped and will be incorrectly diagnosed as wireless losses.

In summary, the congestion loss rate \(p_c\) must be less than \((1/k)\) for \(A_c\) to reach 1. Otherwise, (i.e., if \(p_c \geq (1/k)\)), then \(A_c\) is upper bounded by \(1-(p_c - (1/k))\). Note that \((p_c - (1/k))\) is the fraction of packets marked \textit{in} that would be dropped on average by BQM and incorrectly diagnosed as wireless losses:

\[ A_c \leq 1-(p_c-1/k) \text{ when } p_c \geq 1/k \]  

(5.4)
**Upper bound on k:**

Let $p_c$ be the congestion packet loss probability experienced by a TCP connection at the bottleneck. If $p_c > (1/k)$, then there are more packets dropped due to congestion than the number of arrivals (at the bottleneck) of $out$ packets. Therefore, the router is forced to drop packets marked $in$. When a router starts dropping packets marked $in$, the dropping is no more biased, but rather random. Then, given that a loss is due to congestion, the probability that it was a packet marked $in$ is equal to $1-(1/k)$. If a packet marked $in$ is lost due to congestion:

$$F(x, r, k) = F(0, 1, k) = 1$$

(5.5)

wrongly implying that it is a wireless loss. Therefore, $A_c \leq 1/k$ when $p_c \geq 1/k$. In other words, if $k$ is too large, $A_c$ will be very small. $k$ should be chosen such that $k \leq (1/p_c)$. For a congestion dropping probability of 10% ($p_c = 0.10$), $k$ must be smaller than or equal to 10.

**Impact of the Bottleneck Buffer Size:** If the buffer size at the bottleneck is too small, the biased queue management will not work well because there will not be many packets marked $out$. With a small buffer, packets marked $out$ quickly get exhausted, forcing the drop of packets marked $in$.

**5.2 REACTION TO WIRELESS LOSSES**

Existing work advocates that TCP should not halve the congestion window size or adjust the slow start threshold when a packet loss is known to be due to wireless errors. Even with a perfect loss discriminator, TCP should somewhat throttle its sending rate for wireless losses. TCP should decrease the congestion window size because a wireless loss signals to some extent a temporary decrease of the link layer good put of the wireless
link. Higher the wireless packet loss rate, lower will be the good.put of a wireless link. This decrease may lead to queue build up and congestion drops. Therefore, TCP should at least decrease by one packet its congestion window size for each wireless loss detected (additive decrease). Let us now consider a loss discriminator that is not perfect (i.e., $A_c < 1$ and $A_w < 1$). Let $n$ be the number of packets sent by a TCP sender. For simplicity, retransmissions are neglected. The average number of congestion losses is $n_p_c$. The estimated number $M_c$ of misclassified congestion losses (congestion losses diagnosed as wireless losses) is equal to $n_p_c (1 - A_c)$. Therefore, TCP wrongly does not halve its congestion window size for $M_c$ losses.

The estimated number $M_w$ of misclassified wireless losses is equal to $n_p_w (1 - A_w)$. Here, TCP wrongly halves its congestion window for $M_w$ losses. Optimistically, the wireless and congestion loss misclassifications statistically will cancel each other. However, it is better to have; $M_w \geq M_c$ i.e., the TCP sender should wrongly throttle its sending rate more often than wrongly maintain its sending rate. This requires the term $(M_w / M_c)$ to be known.

Since $(M_w / M_c) = (p_w/p_c) (1 - A_w)/(1 - A_c)$ depends on $p_w$ and $p_c$ that are unknown, $(M_w / M_c)$ must be estimated. Suppose that $n_c$ losses are classified as congestion losses, then a good estimate $n'_c$ of the real number of congestion losses is $(n'_c / A_c)$. Similarly the real number of wireless losses can be estimated as $n'_w = (n'_w / A_w)$. By simple algebraic manipulation, $(M_w / M_c) = (n'_w / n'_c) (A_c/A_w) (1 - A_w)/(1 - A_c)$. Now, suppose that $(M_w / M_c) < 1$ (there are more misclassified congestion losses than misclassified wireless losses). In such a case, TCP congestion control stability may be jeopardized. To avoid this, the only way is to decrease the congestion window also for losses diagnosed as wireless losses.
Further study is needed to determine how to decrease the window size for losses diagnosed as wireless losses such that the effects of misclassifying congestion losses and wireless losses cancel each other statistically. In all simulation experiments of this work, a TCP sender does not back off when a wireless loss is detected.

5.3 EXPERIMENTS

The impact of the value $k$, the round-trip time, and the buffer size on the accuracies $A_c$ and $A_w$ is investigated. This section presents the topology used for the simulations, the packet loss model, and the method used to collect the data. Fig. 5.2 shows the topology used. There are three types of pairs of sender-receiver: TCP connections over type A sender-receiver pair experience the longest propagation delay path with a wireless last hop. There are five routers $R_i$. The dashed lines show the TCP transfers between senders and receivers. With this topology, a competing TCP traffic with different round-trip times is maintained. Bit rates on all links are set such that the bottleneck is the link $R3 - R4$. All senders are TCP senders. Sources are fed with FTP traffic. Varying the bit rate $B_w$ in the bottleneck link $R3 - R4$ and the number of senders $N1$, $N2$ and $N3$ controls the congestion level. Every link is marked with a pair (bit rate, propagation time). While simulations are run with different bit rates, the results presented in this thesis have a bit rate $B_w$ on the wired bottleneck of 45 Mbps and a bit rate $B_{ws}$ on the wireless link of 10 Mbps. Different experiments were run with $N1$,$N2$ and $N3$ varying from 0 to 32. For some experiments, only senders of type 'A' were used to get low congestion packet loss rates.
For the wireless packet loss model, a two-state Markov model is used. In each state, the time between successive losses is exponentially distributed with a mean that depends on the state: in the good state, the mean time is much higher than the mean time for the bad state. The transition probability from the good (resp. bad) state to the bad (resp. good) state is 0.10 (resp. 0.90).

Each experiment lasts 180 seconds. Starting times of the TCP connections are randomly scheduled within a period equal to the round-trip propagation time. Congestion accuracies $A_c$ is collected for each experiment. Note that the same starting times are used to conduct the experiment with normal TCP and proposed TCP. Thirty runs of the same experiment were run, changing only the starting times. The results reported here for $A_c$ are the averages over the 30 runs. As a sample output Table 5.1 shows the impact of $k$ on the discriminator congestion accuracy for different values wireless packet loss rates.
Table 5.1 Impact of $k$ in the discriminator

<table>
<thead>
<tr>
<th>Sl.No</th>
<th>$K$</th>
<th>$p_w=.01$</th>
<th>$p_w=.001$</th>
<th>$p_w=.0001$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>.97</td>
<td>.98</td>
<td>.98</td>
</tr>
<tr>
<td>2</td>
<td>10</td>
<td>.95</td>
<td>.94</td>
<td>.94</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>.82</td>
<td>.80</td>
<td>.80</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>.72</td>
<td>.65</td>
<td>.65</td>
</tr>
<tr>
<td>5</td>
<td>25</td>
<td>.55</td>
<td>.56</td>
<td>.62</td>
</tr>
<tr>
<td>6</td>
<td>30</td>
<td>.50</td>
<td>.51</td>
<td>.59</td>
</tr>
</tbody>
</table>

5.4 CONCLUSION

The problem of distinguishing congestion losses from random wireless losses can be solved by ambiguities in discriminating congestion losses using a biased queue management. This enables the design of an efficient discriminator: The proposed discriminator achieves high accuracies in diagnosing congestion losses and random wireless losses. The conditions under which it is worth distinguishing congestion losses from random wireless losses are identified. The value of $k$ can be chosen dynamically for long-lived TCP connections: at the beginning of a connection, the $k$ value could be set to a small value such that $A_c$ is close to 1 and $A_w$ close to $1-(1/k)$. A small value of $k$ guarantees that the scheme works for high congestion packet loss rates. Under these conditions, a good estimate of $pc$ can be made, leading to the choice of a more appropriate value for $k$ with $k \leq (1/p_c)$. TCP source in general follows the Linear Increase and Multiplicative Decrease (LIMD) model for congestion control which has to take care of the efficiency and fairness. But it converges to the equilibrium state slowly. The next chapter deals with the enhancement of LIMD with Fast Convergence.