Volume: 16 Issue: 04 Pages: 6505-6514 (2025) ISSN: 0975-0290

TCP EWR with Enhanced Bandwidth Estimation to Achieve Improved Performance

Mohammad A. Huq
Dept. of Computer Science and Engineering, Primeasia University, Dhaka, Bangladesh
ariful.huq@primeasia.edu.bd

ABSTRACT

TCP performance over wireless networks often degrades because it cannot effectively differentiate between wireless losses and congestion losses. In this paper, we propose a new version of the TCP protocol, called TCP EWR, which enhances TCP Westwood+ performance by integrating three improvements: in bandwidth estimation, error recovery mechanisms, and loss differentiation techniques. We selected TCP Westwood+ as our base protocol due to its controllable friendliness compared to other protocols. Normally, TCP Westwood+ estimates bandwidth from the sender's side by measuring the rate of returning acknowledgments (ACKs). However, in our approach, we calculate bandwidth using the packet arrival rate, which provides a more accurate estimate in both wired and mixed wired-wireless networks. For error recovery, we developed a more effective scheme that performs better in networks suffering from significant wireless losses. To differentiate between types of losses, we incorporate the Vegas predictor, which helps distinguish between wireless and congestion losses. This paper first analyzes TCP Westwood+ and then details the key techniques used by TCP EWR. We present the results of comprehensive experimental performance studies using simulations, comparing our proposed protocol to major existing protocols such as TCP New Reno, TCP Vegas, TCP Westwood+, and TCP New Jersey. The main finding is that TCP EWR achieves substantial improvements in throughput and goodput over TCP Westwood+, particularly in wired networks. In hybrid wired-wireless networks, TCP EWR also demonstrates better friendliness toward TCP New Reno than TCP Westwood+.

Keywords: **TCP Westwood+, bandwidth estimate, congestion control.**

Date of Submission: October 20, 2024 Date of Acceptance: December 07, 2024

1. Introduction

The Transmission Control Protocol (TCP) is the most widely used transport protocol for Internet applications due to its robustness and reliability. TCP was originally designed for wired networks, where it has demonstrated superior performance. However, with the rapid growth of wireless and mobile networks, communication over wireless links is becoming increasingly prevalent. In wired networks, random bit errors are negligible, and packet loss is generally interpreted as a sign of congestion. In contrast, in wireless networks, most packet losses are caused by high bit error rates during transmission . These adverse channel conditions and periodic disconnections are typically transient phenomena, making the traditional TCP congestion control response both inappropriate and undesirable in such environments. This leads to a sharp drop in throughput and results in underutilization of the network.
Congestion control in TCP has been extensively studied, and many solutions have been proposed to improve TCP performance over wireless networks. However, only a few of these solutions are feasible for practical implementation. Existing solutions can be categorized into three main approaches: End-to-End (E2E) solutions, Split Connection, and Link Layer solutions .

- **End-to-End solutions** involve the sender performing loss recovery techniques. Protocols in this category prevent an immediate window size reset when packet loss is detected, enhancing performance.
- **Link Layer solutions** handle bit errors and packet loss at a lower network layer. They typically combine Automatic Retransmission Request (ARQ) and Forward Error Correction (FEC) techniques , allowing the link layer to address wireless transmission errors. By handling these issues at the link layer, TCP does not perceive wireless-related losses, thus avoiding unnecessary window size reduction.

- **Split Connection protocols** divide the End-to-End connection into two segments: one for the wired part and another for the wireless part. The standard TCP design is used for the wired segment, while a modified TCP, optimized for wireless transmission, is employed for the wireless segment.

TCP EWR, an enhanced version of TCP Westwood+ (TCP WR) , falls under the End-to-End category. In this approach, the receiver measures the packet arrival rate and takes action based on the network's buffer state. Older versions of TCP, such as Tahoe and New Reno , suffer from significant throughput degradation over wireless channels, particularly during random or sporadic losses. TCP EWR addresses these issues by using a time-stamp-based bandwidth estimation approach measured from the receiver's side, rather than relying on conventional bandwidth estimation methods. By integrating this approach with an enhanced error recovery mechanism and loss prediction strategy, TCP EWR achieves significantly improved performance in wireless networks.

The structure of this paper is as follows: Section II provides an overview of the TCP protocols considered, Section III focuses on the key aspects of TCP EWR, Section IV outlines the proposed mechanisms for TCP EWR, Section V presents the simulation results, and Section VI discusses conclusions and future work.

2. Related Works

2.1. TCP New Reno

TCP New Reno is a slight modification of TCP Reno, offering improved performance in the presence of multiple packet losses. While New Reno, like Reno, enters fast retransmit upon receiving multiple duplicate ACKs, it differs by not exiting fast recovery until all outstanding data (i.e., data sent before fast recovery) has been acknowledged. This feature makes New Reno more efficient than Reno in handling multiple losses. However, it suffers from a drawback: detecting each packet loss requires one round-trip time (RTT). The protocol can only deduce additional lost segments once the acknowledgment (ACK) for the first retransmitted segment is received.

2.2. TCP Vegas

TCP Vegas extends TCP Reno's retransmission mechanism with key improvements, particularly in the congestion avoidance phase, where it does not rely on segment loss to detect congestion. The flow control scheme of TCP Vegas introduces three key modifications. Vegas requires the sender to record the round-trip time (RTT) and timestamp when each packet is sent. By comparing its actual sending rate with the expected rate, Vegas estimates the network's congestion state. This approach helps TCP Vegas avoid overshooting link bandwidth, making it more efficient in managing congestion.

2.3. TCP Westwood+

TCP Westwood+ is a sender-side modification of TCP New Reno, designed to better handle high bandwidth-delay product paths, dynamic loads, and packet losses due to transmission errors. TCP Westwood+ evolved to address the issue of ACK compression in TCP Westwood. It estimates available bandwidth by counting and filtering the flow of returning ACKs, then adaptively adjusts the congestion window (cwnd) and slow-start threshold (ssthresh). Unlike TCP Reno, which blindly halves the congestion window upon receiving three duplicate ACKs, TCP Westwood+ sets cwnd and ssthresh based on the estimated available bandwidth at the time of congestion. This approach prevents overly conservative reductions, leading to faster recovery and more efficient use of the network.

2.4. TCP New Jersey

TCP New Jersey is capable of distinguishing between packet losses caused by wireless errors and those caused by congestion, allowing it to react accordingly. It incorporates two key components: the Timestamp-Based Available Bandwidth Estimation (TABE) algorithm and the Congestion Warning (CW) router configuration. TABE is a sender-side algorithm that continuously estimates the bandwidth available to the connection and guides the sender in adjusting its transmission rate during congestion. It is immune to issues like ACK drops and compression. The CW component involves network routers marking packets to signal end stations when congestion is imminent, allowing for timely adjustments to avoid severe congestion.

2.5. FAST TCP

FAST TCP introduces a log utility function and achieves weighted proportional fairness. It differs from traditional TCP in three significant ways:

1. Its window adjustment is based on equations, eliminating packet-level oscillations.
2. It uses queuing delay as the primary measure of congestion, providing a more accurate and scalable measure of congestion with network capacity.
3. It has stable flow dynamics, achieving weighted proportional fairness at equilibrium without penalizing long flows, unlike current congestion control algorithms.

3. Bandwidth Estimation in TCP Westwood+

This section describes the congestion control algorithm of TCP Westwood+ and the challenges associated with bandwidth estimation. Specifically:

- **Section 3.1** outlines the algorithm,
- **Section 3.2** summarizes the bandwidth estimation technique,
- **Section 3.3** discusses the disadvantages of the current bandwidth estimation approach.

3.1 Algorithm for TCP Westwood+

Recently TCP Westwood+ uses following Algorithm[1] for Congestion Control:

```
i. When ACKs are successfully received:
        cwnd increases following the Reno algorithm
ii. When 3 DUPACKS are received
        ssthresh =(BWE*RTTmin)/seg_size
        cwnd = ssthresh
iii. When a coarse timeout expires
        ssthresh=(BWE* RTTmin)/seg_size
        cwnd =1
```

It explains that when three duplicate ACKs (DUPACKs) are received, both the congestion window (cwnd) and the slow start threshold (ssthresh) are set to the product of the estimated bandwidth (BWE) and the minimum measured round-trip time (RTTmin). When a coarse timeout occurs, the ssthresh is set in the same way, but the cwnd is reset to one.

In general, TCP Westwood+ follows an Additive Increase/Adaptive Decrease approach. This means that the congestion window (cwnd) increases additively when the sender receives ACKs, and both the cwnd and ssthresh are adjusted adaptively in response to signs of congestion.

3.2 End-to-end Bandwidth Estimate for TCP Westwood+

If an ACK is received at the sender at time t_k and within this time amount of data, d_k that has been received by the receiver therefore, we can measure the sample BW:

$$b_k = \frac{d_k}{t_k - t_{k-1}} \qquad (1)$$

Here, t_{k-1} is the time that previous *ACK* was received. Generally congestion occurs when low frequency traffic rate crosses the capacity level so a low pass filter has been proposed [4] to obtain low frequency components of the of the available bandwidth. Now using Tustin approximation [4] we have:

$$\hat{b}_k = \frac{\frac{2\tau}{t_k - t_{k-1}} - 1}{\frac{2\tau}{t_k - t_{k-1}} + 1}\hat{b}_{k-1} + \frac{b_k + b_{k-1}}{\frac{2\tau}{t_k - t_{k-1}} + 1} \qquad (2)$$

where $\hat{b}_k$ is the filtered measurement of the available bandwidth at time $t = t_k$, and $1/\tau$ is the cut-off frequency of the filter. TCP-Westwood+ also includes a timer in its bandwidth estimator if time τ/m ($m>2$) has passed without receiving any new ACKs, the filter assumes the reception of a sample. When TCP-Westwood+ determines that the link is congested after receiving three dupacks, it sets the SS threshold to reflect its estimated bandwidth delay product as :

$$ssthresh = \frac{BWE \times RTT\min}{seg\_size} \qquad (3)$$

3.3. Problems in Westwood+

In TCP Westwood+, bandwidth (BW) estimation is based on ACK information. When the sender needs to retransmit segments, additional time is spent on retransmission processing, which can lead to degraded performance. If the number of arriving ACKs is smaller than expected, the results will be poor. Additionally, RTTmin is used to calculate the slow start threshold (ssthresh), and it becomes critical to accurately handle the RTT value, especially when the network is congested.
In wireless channels, the reverse link may suffer from transmission errors, which conventional bandwidth estimation techniques cannot measure. This results in issues such as ACK compression, ACK delay, and ACK losses, all of which significantly impact the overall performance of the TCP scheme. If rate control is performed during the slow start (SS) phase, the ssthresh will typically be set higher, allowing the available bandwidth to be reached more quickly. Furthermore, if a retransmission timeout (RTO) occurs during the SS phase, the ssthresh will not be halved as it is in TCP Reno or TCP Jersey.
ACK values play a key role in rate control, but the transmission path to the receiver and the return path for the ACKs may differ, leading to varying delays depending on factors such as router buffer queuing. Additionally, if ACK drops or compression occur, the ACK rate loses its significance due to the insufficient sample size, further complicating the bandwidth estimation.

4. Proposed Approach

4. 1 BW Estimation approach for TCP EWR

In order to solve the ACK compression effect from TCP Westwood+, timestamp should be used or an additional mechanism reflecting only valid *ACK* values are necessary. In RFC 1323[8] timestamp option was proposed to achieve reliability of the available BW estimation when we consider a network which transmits bi-directional traffic.
The basis of, measuring BW of TCP EWR will be packet arrival to receiver instead of *ACK* arrival to the sender. This is a rate-based congestion window control procedure where every packet that arrives at the receiver is marked by a timestamp that is sent back by the *ACK*. This is equivalent to having the estimation done at the receiver; sending feedback information to the receiver is not at all necessary here.

In our previous work of TCP WR, the sample bandwidth what we considered is as follows:

$$Bw_{sample}(n) = \frac{p_n}{t_n - t_{n-1}} \qquad (4)$$

where $Bw_{sample}(n)$ is the sample bandwidth when the n_{th} packet reaches to the receiver at time t_n, Again t_{n-1} is the previous packet arrival time. Here, for estimating packet arrival rate we did not consider packet loss probability and for this reason it cannot maintain appropriate sending rate when the network suffers packet loss either by network congestion or wireless high bit-error-rate (BER). Now in this work of TCP EWR, sample bandwidth estimation has been computed more precisely by inserting packet loss rate information. We used the renowned model of Padhey et al [10] where $RTT(n)$ is the estimated round trip time for nth packet, $l(n)$ is the probability of packet loss which

$$p_n = \frac{1}{RTT(n)}\sqrt{\frac{3}{2l(n)}} \qquad (5)$$

assumes congestion-related packet losses. By accounting loss rate information in packet arrival, network can probe the available bandwidth more accurately.

TCP EWR computes current available bandwidth through a time varying coefficient, Exponentially Weighted Moving Average (EWMA) Filter. The Estimation of the bandwidth can be expressed as follows:

$$Bw_{estimated}(n) = Bw_{estimated}(n-1)\times\beta_n + Bw_{sample}(n)\times(1-\beta_n) \qquad (6)$$

Here β_n is a constant filter-gain, in our simulations we set this value to .7.

With the help of the literature [4] the optimal congestion window $oCwnd_n$ has been calculated which is expressed as in unit of segments:

$$oCwnd_n = \frac{RTT_{\min} * Bw_{estimated}}{Seg\_sige} \qquad (7)$$

Where *RTT* is the round trip time, Seg_size is the length of the TCP Segment.

Furthermore, our simulation results show that the proposed formula also provides accurate bandwidth estimation in scenarios with random packet loss.

4.2. Loss Detection

TCP Westwood+ uses a reactive approach to congestion detection and control, where segment loss is interpreted as an indication of congestion in the network. However, it lacks a mechanism to detect the early stages of congestion before losses occur, which could help in preventing them. Once the connection reaches the congestion avoidance phase, the congestion window (cwnd) increases linearly until congestion is detected, at which point it drops to match the available bandwidth. This cycle repeats for the duration of the connection.

Several studies [21] have discussed the pros and cons of various approaches proposed for proactive congestion detection, including Vegas Predictor, NTZ, and NTG. These approaches aim to detect congestion early based on an understanding of network changes. It has been shown that, when packet loss is accurately diagnosed, the Vegas Predictor performs more efficiently than other methods. In our work, we adopt the Vegas Predictor concept [13] to differentiate between losses. It measures the outstanding packets on the network by comparing the expected and actual bandwidth. The Vegas Predictor estimates the maximum expected transmission rate and measures the actual transmission rate whenever an ACK is received. A variable, D, is calculated as the difference between the actual and expected transmission rates.

$$\text{Expected Throughput} = \frac{WindowSize}{BaseRTT} \quad (8) \text{ and}$$

$$\text{Actual throughput} = \frac{WindowSize}{RTT} \quad (9)$$

Then the difference *D* is calculated as,

D = Expected throughput - Actual throughput

The value of D presents the amount of current existing data in buffer of network nodes. Namely, D indicates the state of the current network. This is given by:

$$x = BaseRTT \times D = WindowSize\left(1 - \frac{BaseRTT}{RTT}\right) \qquad (10)$$

It also defines two thresholds, α and β, roughly corresponding to having too little and too much extra data in networks. When x < α, TCP EWR increases the congestion window linearly during the next *RTT*, and when x > β, it decreases the congestion window adaptively during the next *RTT*. TCP and leaves the congestion window unchanged when α < x < β. We assumed the *minRTT* value as 9.0ms and the value of α and β were considered as 1 and 9 respectively.

4.3 Enhanced Error Recovery (EER)

TCP EWR will activate the Enhanced Error Recovery (EER) mechanism if the network encounters a critical timeout expiration. Under normal conditions, TCP EWR follows the congestion control mechanism of TCP Westwood+. However, if there is any indication of packet loss, either due to RTO expiration or receiving 3 duplicate ACKs, TCP EWR will adjust the **ssthresh** accordingly.

In the event of a bit error rate (BER), the total loss period (denoted as Wireless_Loss_Time) is first calculated using the timestamp option, and this value is divided by a standard RTT. For simplicity, we assume minRTT in this case. Based on the number of W_rtt intervals, the congestion window (cwnd) will be increased by **ssthresh** with each step, prioritizing the erroneous node. Although this is an aggressive strategy, it yields positive results in networks severely affected by high wireless BER, allowing for more efficient bandwidth utilization

```
if (RTO Expired)
    if (x < $\alpha$)  /* loss due to congestion*/
```

```
            ssthresh = $oCwnd_n$ ;
          if (ssthresh < cwnd)
                cwnd = $oCwnd_n$ ;
          end if
      else  /* if (x > β), if loss due to BER*/
            ssthresh = $oCwnd_n$
        W_rtt = Wireless_Loss_Time / minRTT ;
                                    /*in integer value  */
        for(i=0; i<W_rtt; i++)
        {
           cwnd = ssthresh;
        }

      else
            cwnd = cwnd;      /* No Change */
      end if
  end if
```

Fig. 1: Error Recovery of TCP EWR

4.4 Congestion Control Algorithm of TCP EWR

The congestion control algorithm of TCP EWR is:

i. When Packets are successfully received in the Receiver:

  cwnd = *cwnd* + 1;

ii. When 3 DUPACKS will be sent:

ssthresh =(BWE*RTTmin)/seg_size

  cwnd = *ssthresh*

iii. When a coarse timeout expires

  if (Check for Wireless Error)

   EER Approach

  else

   ssthresh=(BWE* RTTmin)/seg_size   cwnd =1

5. Performance Metrics:

Throughput, goodput, fairness and friendliness are now considered as important metrics to measure the TCP performance on wired or wireless network.

5.1. Throughput

The aggregate rate of packets generated by all sources is depicted as throughput. An analytical model for TCP, (PFTK model) was developed by J. Padhye et. al.[10] which shows mathematical model for expected TCP throughput R as a function of several path characteristics For instance, R = f (T, p, W, A) Path characteristics, as experienced by target TCP flow:

$$R = \min\left(\frac{M}{T\sqrt{\frac{2bp}{3}} + T_0 \min(1, \sqrt{\frac{3bp}{8}})p(1+32p^2)}, \frac{W}{T} \right), p > 0 \tag{11}$$

Here

- T: *RTT*
- p: loss rate
- M: path MTU (Maximum Transfer Unit)
- W: TCP maximum window (due to socket buffer size limits)
- T_0: TCP retransmission timeout
- b: number of segments per ACK (b=2)

The TCP throughput also can be obtained combining the losses and the RTTs using the Mathis formula [11] for deriving the maximum TCP throughput:

$$\text{Throughput} = \text{MSS}/(RTT*\text{sqrt(loss)}). \tag{12}$$

5.2. Goodput

Goodput is regarded as the rate at which packets arrive at the receiver. Goodput differs from throughput in that retransmissions are excluded from goodput. It is an important metric that shows useful application data successfully is processed by the receiver. Throughput consisting of useful traffic measured in bytes per second is defined as goodput [15]:

$$\text{Goodput} = \frac{\text{sent data- retransmitted data}}{\text{transfer time}} \tag{13}$$

5.3. Fairness

Another evaluation for TCP is fairness that a set of connections of the same TCP, which can share fairly the bottleneck bandwidth. The index of fairness was defined in [16] as:

$$F(x) = \frac{\left(\sum x_i\right)^2}{\left(m\sum x_i\right)^2} \tag{14}$$

where *xi* is the goodput of the *i*-th connection and *m* denotes the number of connections sharing the bottleneck. The Jain fairness index belongs to the interval [1/*m*, 1] and increases with fairness reaching the maximum value at one.

5.4 Friendliness

The friendliness of TCP implies fair bandwidth sharing with the existing TCP versions. The notion of TCP-friendliness refers to the relationship between throughput and packet loss rate. “TCP-Friendliness” indicates that the protocol chooses to send at a rate no higher than TCP under similar conditions of round trip delays and packet losses.

6. Simulation Results

From NS-2[18] Simulation, we tried to compare our proposed estimate with the various TCP protocols in terms of the above described matrices. Westwood+ NS-2 Module is available in the internet [19].

6.1. Throughput Comparison over Lossy Link (Simple Scenario)

We ran our simulation for the lossy link environment. We used FTP over TCP and CBR over UDP with the same packet size of 1000 bytes. We considered the bottleneck link bandwidth as 5Mbps and delay was 5ms.The link error

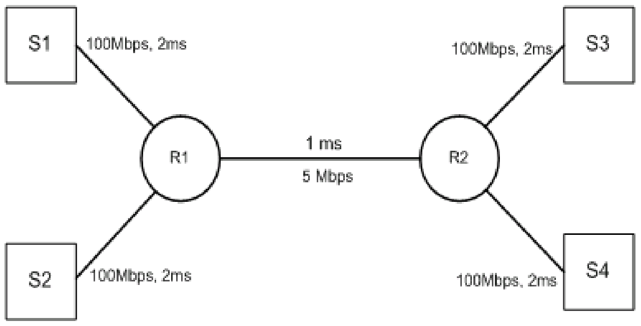

Fig. 2. Simulation Scenario for Forward Traffic

rate was ranging from .01% to 70% and simulation time was 325 sec. Fig 2 reveals the experimental setup for our experiment.

From this figure we can assume that our proposed protocol, TCP EWR can achieve improved performance

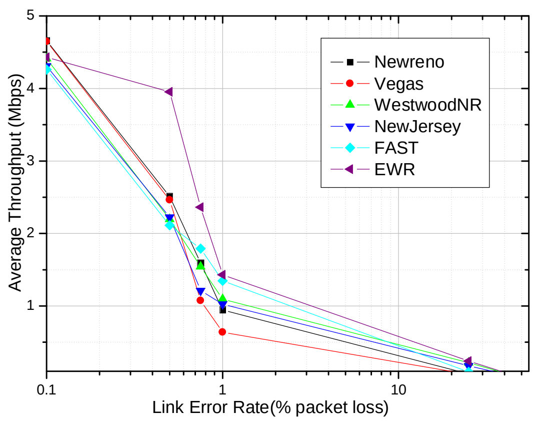

Fig. 3. TCP Throughput vs. lossy link Error Rate

when link error rate becomes stronger. In low error rate its performance is identical like other protocols. When error rate is increased more than .1% we can see the significant improvement in throughput comparing to other protocols. On further increase on error rate, TCP EWR always achieved relatively better performance than New Reno, Westwood+, New Jersey, FAST and Vegas.

6.2. Network topology with Reverse Traffic

Now considering the network topology as shown in figure 4, throughput of TCP EWR will be compared with other TCP versions in the presence of background traffic on various link error rates. Here FTP application is considered, where initial window size is 2000 and bottleneck link bandwidth is 5Mbps and delay is 1ms. The node N1 and N2 connects to Node R1 via a 100MB wired link with 2 ms propagation delay. R1 is connected to R2 via a 5MB wired link with 1 ms propagation delay. Similarly, the node N3 and N4 connects to Node R2 via a 100MB wired link with 2 ms propagation delay. The cross traffic flow is FTP background traffic via 100MB wired link with 2 ms propagation delay. Queue size of the bottleneck link is set to 3000 and simulation time was 21sec.

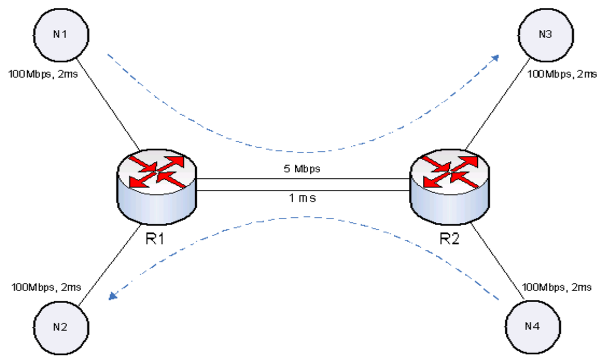

Fig. 4. Topology with Reverse Traffic

The throughput comparison result of the FTP bi-directional background traffic is presented in figure: 5. In low link error rate TCP EWR shows identical performance like other TCP versions. When link error rate reached .1% at that time TCP EWR outperforms TCP New Reno by 2%, TCP Vegas by 34%, TCP Westwood+ by 8% and TCP New Jersy by 15%. In 1% error rate performance of TCP EWR increases considerably than other TCP

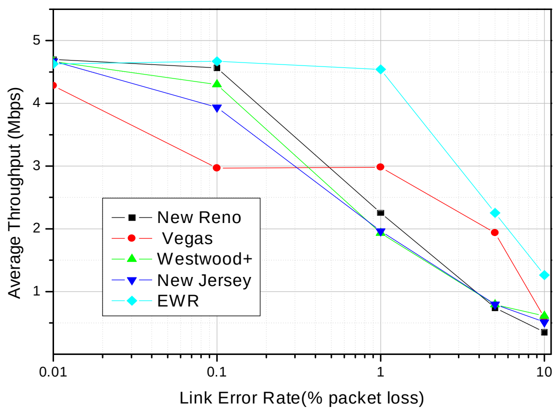

Fig: 5. Throughput Comparison

versions. It can be seen from the figure that here TCP EWR outperforms TCP New Reno by 45%, TCP Vegas by 32%, TCP Westwood+ by 52% and TCP New Jersy by 51%.

Even, in 10% error rate where other TCP version's throughput has been very low but TCP EWR responds well in this high error prone environment. It can be deduced that TCP EWR has significant performance improvement in erroneous network environment.

6.3. Complex Network topology

In figure 6, a more complex topology has been chosen to measure goodput and throughput of TCP EWR and after that these estimates has been compared with other TCP versions on various link error rates in the presence of background traffic. Here 10 nodes from each Router (R1, R2) side have been attached with 100MB wired link with 2 ms propagation delay. Bottleneck link bandwidth is 10 Mbps and delay has been set to 1ms.

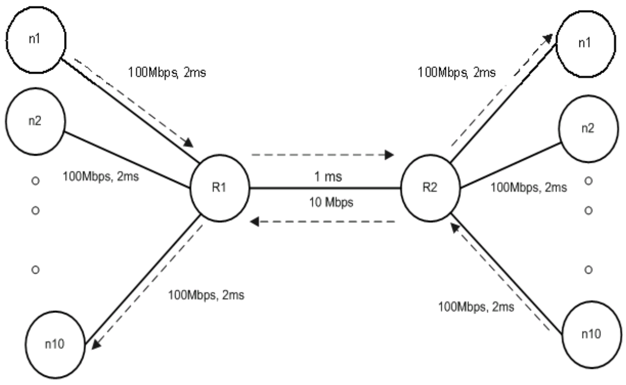

Fig. 6. Simulation scenario (A larger View)

Considering the topology shown in figure: 6 we have compared throughput of TCP EWR with other major TCP versions in figure: 7. It can be seen that TCP EWR shows

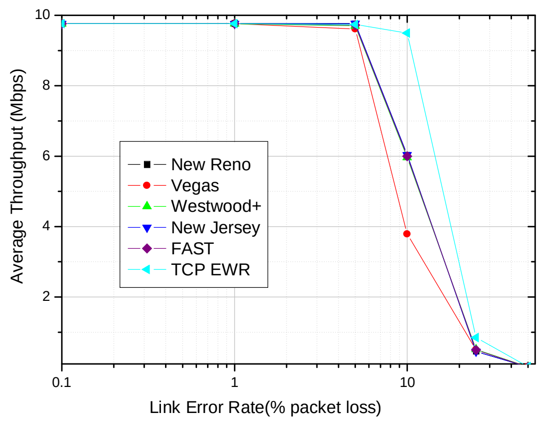

Fig: 7: Throughput Comparison

identical performance when link error rate is low. On further increase in error rate i.e. at 8% TCP EWR outperforms all of the TCP versions. When error rate increases more, TCP EWR shows considerable lead than other TCP versions.

Figure:8 represents the result of Goodput comparison, which considers the same scenario of Figure: 6, we know goodput actually indicates the measurement of available bandwidth estimation. From the graph it is evident that TCP EWR shows identical performance when link error rate is low. Actually the result of goodput comparison strongly resembles the throughput comparison's result which has been shown in figure: 10. For error rate, more than 8% it is clear that TCP EWR outperforms all the TCP versions. When error rate increases more TCP EWR maintains significant lead than other TCP versions.

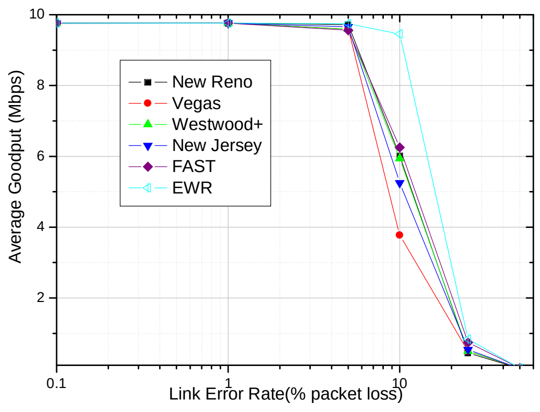

Figure 8: Goodput Comparison

6.4. Fairness Comparison

Considering the topology shown in figure:8 and using Jain fairness index from equation: 13, TCP fairness has been calculated where total 10 same TCP flows have been

Error Rate	New Reno	Westwood+	EWR
0	0.76	0.80	0.64
0.1	0.31	0.93	0.98
1	0.25	0.74	0.76
10	0.57	0.23	0.62

Table:1 : Fairness Comparison

considered those are sharing 10Mbps bottleneck link. Different TCP versions have been simulated individually and comparison has been shown in table: 1. In fairness perfectly fair bandwidth allocation results a fair index of 1. Here link error ranges from 0% to 10%. We can see that except 0% error rate TCP EWR's fairness index achieves a satisfactory margin comparing other protocols.

7. Simulation Results for Wired Cum Wireless Network

7.1 Simple Network topology

In figure 9 a simple network topology has been considered where the source (Node S) connects to Node

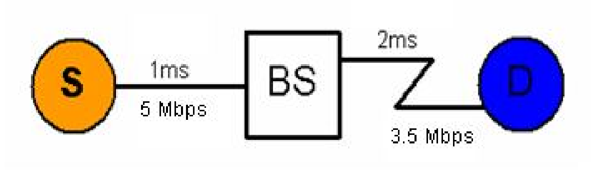

Figure 9: Simple Network Topology

BS via a 5 Mbps wired link with 1ms propagation delay. Node BS is linked to the destination (Node D) via a 3.5Mbps wireless link with 1ms propagation delay.

7.2 Goodput Comparison

Figure: 10 represents the result of Goodput comparison for single TCP flow from figure 9. Here it can be seen that, TCP EWR is showing a better performance from the starting point. TCP New Reno shows identical performance here but TCP EWR has kept a slight lead over TCP Reno in most of the cases. Comparing other TCP version like Vegas and Westwood+ TCP EWR has shown better performance throughout the whole period but the lead is moderate.

7.3. Complex Network topology

In figure 11 multiple TCP connection has been considered where the source (Node S) connects to Node R1 via a 100MB wired link with 10ms propagation delay. R1 is linked to Node BS via a 100MB wired link with 1ms propagation delay. The asymmetric wireless link from BS to the destination (Node D) is represented by the differing bandwidth on the 5MB and 1ms propagation delay. The bidirectional traffic flows, Node N1 to N2 and N4 to N3 are FTP background traffic via a 100MB wired link with 1ms propagation delay. The queue size of the wired link is set to 100 and the wireless link queue size is set to 10.

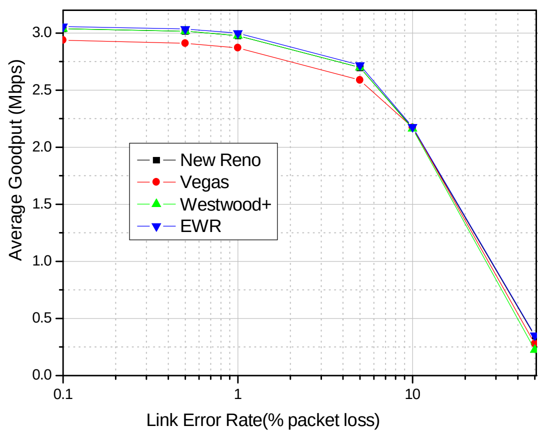

Figure 10: Goodput Comparison

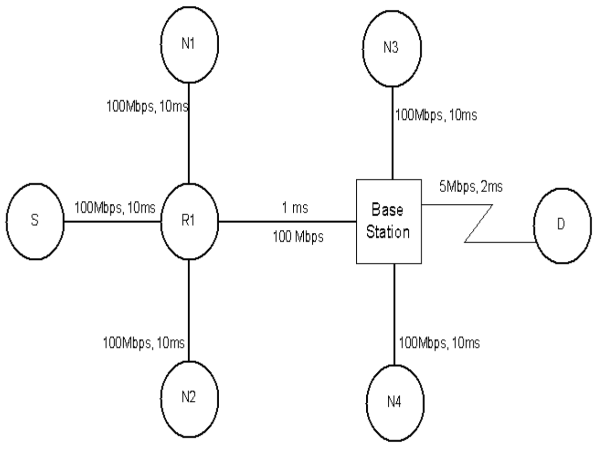

Figure 11: Complex Network Topology

linked to Node BS via a 100MB wired link with 1ms propagation delay. The asymmetric wireless link from BS to the destination (Node D) is represented by the differing bandwidth on the 5MB and 1ms propagation delay. The bidirectional traffic flows, Node N1 to N2 and N4 to N3 are FTP background traffic via a 100MB wired link with 1ms propagation delay. The queue size of the wired link is set to 100 and the wireless link queue size is set to 10.

7.4 Throughput Comparison

Considering the scenario of fig: 11, throughput comparison result has been shown in figure 12. In 5% error rate TCP EWR outperforms TCP Westwood+ by 10% and TCP Vegas by 12% but in this case TCP New Reno shows identical performance with TCP EWR. On the error rate after 10% TCP Westwood+ minimizes the throughput loss and rest of the period TCP EWR has shown similar performance like New Reno and Westwood+ performance like New Reno and Westwood+ with maintaining a small lead comparing other protocols. TCP Vegas has shown poor performance throughout the whole period.

7.5 Fairness Comparison

For the topology of the hybrid wired/wireless network shown in figure 11, fairness index has been calculated using Jain fairness index where 7 TCP flows of same versions share a bottleneck link bandwidth 5Mbps. The Result has been summarized in the table: 3.

Error Rate	New Reno	Westwood+	EWR
0	0.95	0.94	0.96
0.1	0.94	0.94	0.89
1	0.95	1.00	0.95
10	0.96	0.95	0.96

Table 2: Fairness Comparison

Here we can see all the TCP versions including TCP EWR achieve fairly satisfactory level of fairness index

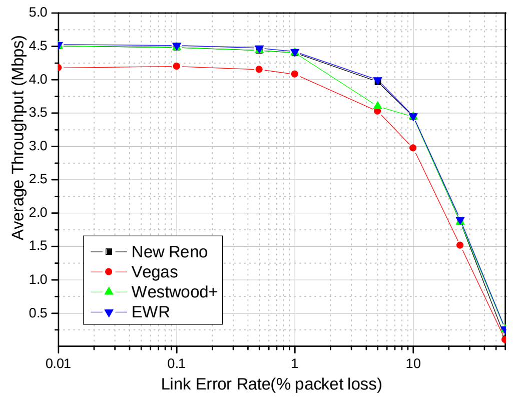

Figure 12: Throughput Comparison

7.6 Friendliness Comparison

To verify the friendliness of TCP EWR a hybrid wired and wireless network topology has been created which is

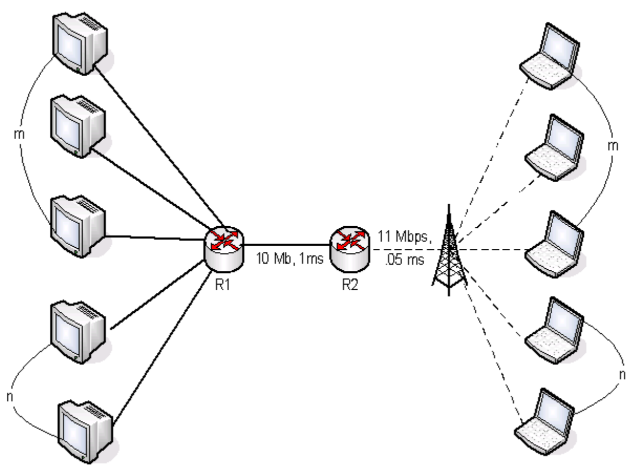

Figure 13: Simulation topology for verifying friendliness

shown in figure: 13. Two major TCP versions TCP New Reno and TCP Westwood+ have been tested here and in table: 4 and 5 the result of coexistence of TCP EWR with these versions has been shown. The wired link bandwidth is 10 Mbps and the delay is 1 ms and wireless link has 11 Mbps and .05 ms delay. 10 pairs of connections in which m are TCP EWR connections and n are other TCP connections. Here link error rate has been assumed to be very high. Average throughput has been calculated in Kb/s.

No of Reno Connections	No of EWR Connections	New Reno: Mean Throughput	EWR: Mean Throughput
3	7	69.24	59.97
5	5	77.91	77.68
7	3	71.55	71.03

Table 3: Friendliness Comparison of TCP EWR with TCP New Reno

No of Westwood+ Connections	No of EWR Connections	Westwood+: Mean Throughput	EWR: Mean Throughput
3	7	73.71	81.95
5	5	78.03	97.03
7	3	63.25	70.62

Table 4: Friendliness Comparison of TCP EWR with TCP Westwood+

From the above tables it is clear that TCP EWR has very good controllable friendliness with other TCP versions in error prone environment.

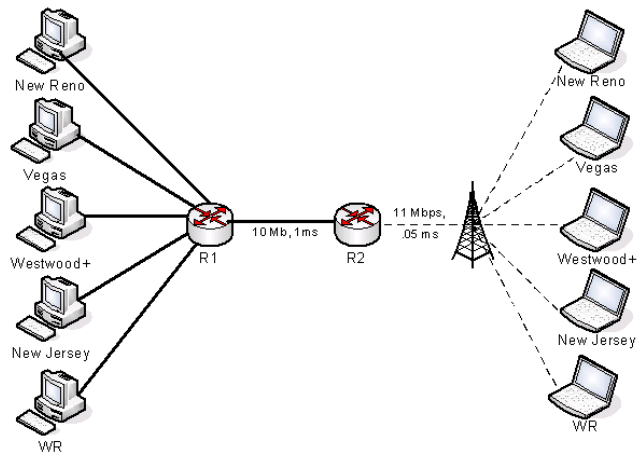

Figure 14 : Simulation topology for coexistence of TCP Scheme

In another scenario, we tested TCP EWR's coexistence with other major TCP versions, and the throughput results are presented in Table 5. It is evident that, apart from TCP Vegas, most TCP versions performed similarly when compared to TCP EWR. Additionally, TCP EWR's friendliness was found to be within the same range as TCP New Reno. Throughout the testing period, TCP EWR demonstrated stable and manageable friendliness with TCP New Reno, while its friendliness with other versions, such as TCP New Jersey and TCP Westwood+, remained at a moderate level.

Error Rate	New Reno	Vegas	Westwood+	New Jersey	EWR
0	1.5528	0.2626	0.8962	0.9954	1.3978
0.1	1.4593	0.2355	0.9190	1.1133	1.3843
10	1.0986	0.2852	0.5397	0.6935	1.2301
25	0.6106	0.2852	0.3005	0.3120	0.6378

Table 5: Coexistence Comparison

8. Conclusion & Future Research

In this paper, we propose a new end-to-end method for a modified TCP version called TCP EWR (Receiver-Side Modification on TCP Westwood+). Our approach introduces a novel bandwidth (BW) estimation technique and an enhanced error recovery mechanism, which together improve TCP performance in both wired and wireless networks. The BW estimation method we describe is more accurate compared to existing approaches. We benchmarked our method against several major TCP versions, utilizing ns-2 modules available online. Our analysis covers both wired and wireless network challenges, and simulation results show that in error-prone environments, TCP EWR outperforms other existing protocols.

In the future, we aim to consider more test cases to address fairness and friendliness issues, alongside a more detailed comparison with existing protocols. TCP EWR can be further refined to detect various types of wireless losses, including those caused by random errors and handoff processes. Additionally, a prediction-based approach for congestion avoidance can be implemented to achieve better throughput and link utilization. We believe that improvements in these areas will further strengthen TCP's viability for wireless data communication.

9. Refferences

[1]. Grieco L. A., Mascolo S. "Performance evaluation and comparison of Westwood+, New Reno and Vegas TCP congestion control" ACM Computer Communication Review, vol. 34, no. 2, April 2004.
[2]. Floyd, S, Henderson, T 1999 FC2582- The New Reno modification to TCP's fast recovery algorithm.Retrieved April 28, 2005,.http://www.faqs.org/rfcs/rfc2582.html
[3]. Mascolo Saverio, Casetti Claudio, Gerla Mario, Sanadidi M. Y. and Wang Ren "TCP Westwood: Bandwidth Estimation for Enhanced Transport over Wireless Links" Proceedings of ACM Mobicom, Rome, Italy, July 2001
[4]. Xu K., Tian Y., Ansari N., "TCP-Jersey for wireless IP communications", IEEE Journal on Selected Areas in Communications 22 (4) (2004) 747–756.
[5]. A brief history of TCP Retrieved 2005. Retrieved from http://www.erg.abdn.ac.uk/research/satcom/tcp-evol.html
[6]. Dhiman Barman, Ibrahim Matta, Eitan Altman, and Rachid El Azouzi. TCP Optimization through FEC, ARQ and Transmission Power Tradeoffs. Technical report, Boston University, Computer Science Department, Boston, MA 02215, 2003.
[7]. Paul Rajashree and Trajković Ljiljana "Selective-TCP for Wired/Wireless Networks" SPECTS 2006, Calgary, AL, Canada, Aug. 2006, pp. 339-346.
[8]. [8] V.Jacobson, Braden R., Borman D. "TCP Extensions for High Performance" RFC 1323May 1992,
[9]. L. Brakmo, S. O'Malley, and L. Peterson. TCP Vegas: New techniques for congestion detection and avoidance. In Proceedings of the SIGCOMM '94 Symposium (Aug. 1994) pages 24-35.
[10]. J. Padhey, V. Firoiu, D. Towsley, and J. Kurose, Modeling TCP throughput: a simple model and its empirical validationProceedings of ACM SIGCOMM 98 vol. 28 (1998) pp. 303–314.
[11]. M. Mathis, J. Semke, J. Mahdavi and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," Computer Communication Review 27,1997.
[12]. S. Floyd, "Metrics for the evaluation of congestion control mechanisms," IETF Internet-draft, August 2006.
[13]. Biaz, S., and Vaidya, N.H.: Distinguishing congestion losses from wireless transmission losses: a negative result. IEEE 7th Int. Conf. Computer Communications and Networks, Lafayette, LA, October 1998, pp. 390–397
[14].J. Postel. RFC 793: Transmission control protocol, Sept. 1981.
[15]. S. Floyd, "Metrics for the evaluation of congestion control mechanisms," IETF Internet-draft, August 2006.
[16]. Jain, R., Chiu, D., and Hawe, W.: A quantitative measure of fairness and discrimination for resource allocation in shared computer systems, Research Report TR-301, (1984)
[17]. Mohammad Ariful Huq, "TCP WR, A Receiver Side Modification of TCP Westwood+ to Achieve Improved Performance", 7th International Conference, Communications 2008, pp 235-238, Bucharest, Romania, June-2008.
[18].http://wwwmash.cs.berkeley.edu/ns ns-2 network simulator (ver.2).
[19].ns-2 Westwood+ implementation Available: http://www.ictserv.poliba.it/mascolo/tcp%20Westwood/modules.htm
[20].C. Jin, D. X. Wei, and S. H. Low, "TCP FAST: Motivation, Architecture, Algorithms, Performance," Proc. IEEE INFOCOM, Mar. 2004, http://netlab.caltech.edu
[21].Saad Biaz, Nitin H. Vaidya: Distinguishing Congestion Losses from Wireless Transmission Losses-A Negative Result. ICCCN 1998: 722-731

Authors Biography

Muhammad Ariful Huq received the B.Sc. degree in computer science and information technology from the Jahangirnagar University (JU), Bangladesh, in 2005, and MS. degree in computer engineering from Kyung Hee University, South Korea, in 2008 and MPHil in Electronic Engineering from Macquarie University, Australia in 2014. He is currently a Faculty Member of the Department of Computer Science and Engineering, Prime Asia university, Dhaka, Bangladesh. His research interests include wireless MAC protocol development, mostly WBAN MAC, including IP and transport layer issues, emergency channel access mechanism and QoS provisioning. He has served as a reviewer in several international conferences/Journal.