TCP-New Veno: The Energy Efficient Congestion Control in Mobile Ad-hoc Networks

Namho Cho and Kwangsue Chung

School of Electronics Engineering, Kwangwoon University, Korea
nhcho@adams.kw.ac.kr and kchung@daisy.kw.ac.kr

Abstract. In recent years, there have been many researches about Mobile Ad-hoc Networks (MANETs) which is available to communicate freely between mobile devices by using multi-hop without any support of relay base or access point. TCP that used the most widely transport protocol in the Internet repeats packet loss and retransmission because it increases congestion window size by using reactive congestion control until packet loss occurs. As a result of this, the energy of mobile device is wasted unnecessarily.
In this paper, we propose TCP-New Veno in order to improve the energy efficiency of mobile device. According to the network state, the scheme adjusts appropriate size of congestion window. Therefore, the energy efficiency of mobile device and utilization of bandwidth are improved by the scheme. From the simulation by using ns-2, we could see more improved energy efficiency with TCP-New Veno than those with TCP in MANETs.
Keywords: Mobile Ad-hoc Networks (MANETs), TCP, Energy efficiency, Congestion control, Packet loss

1 Introduction

In recent years, there have been many researches about Mobile Ad-hoc Networks (MANETs) which is available to communicate freely between mobile devices by using multi-hop without any support of relay base or access point. Unlike wired networks, there are some unique characteristics of MANETs. These characteristics include the lossy wireless channels due to noise, fading and interference, and the frequent route breakages and changes due to node mobility. Moreover, mobile nodes in MANETs can be an important part of MANETs because they can function as not only end-host but also intermediate node. Therefore, the lifetime of mobile nodes is closely related to the lifetime of MANETs [1, 2].

TCP employs reactive congestion control. This congestion control repeats packet losses and retransmissions for recovery of the lost packet. Repeated retransmissions are not the problem in wired networks, but lead to waste the limited battery power of mobile nodes in MANETs. As a result, networks could be collapsed because the lifetime of mobile nodes is closely related to the lifetime of MANETs. Moreover, TCP can not distinguish between congestion loss and link error loss. TCP unnecessarily invokes congestion control when a packet is lost due to link error. Since TCP wastes the limited bandwidth of MANETs, its performance is worse in MANETs.

In this paper, we propose TCP-New Veno in order to improve the performance of TCP in MANETs. According to the network state, the scheme adjusts appropriate size of congestion window. Moreover, when packet loss has occurred, the scheme invokes appropriate congestion control according to a cause of packet loss. Therefore, the energy efficiency of mobile device and utilization of bandwidth are improved by our proposed scheme.

The rest of this paper is organized as follows. Section 2 describes the problem with TCP in MANETs and the proposed work. The following section outlines the principles of our TCP-New Veno. The subsequent section presents the simulation results, and lastly we conclude the work.

2 Related Work

If TCP is used without modification in MANETs, the performance of TCP is seriously dropped. In this section, we describe the problem with TCP in MANETs and the related works.

2.1 The Problem with TCP in MANETs

TCP employs reactive congestion control. The congestion window is adjusted on the basis of the collective feedback of ACKs and DUPACKs generated at the receiver. TCP probes for the available bandwidth by continuously increasing the congestion window size gradually until the network reaches the congestion state. When packet loss has occurred, TCP will then fall back to a much slower transmission rate for reducing the congestion. TCP increases the congestion window size without taking the network capacity into consideration. In this sense, congestion and packet loss is inevitable. Therefore, the congestion control of TCP repeats packet losses and retransmissions for recovery of the lost packet. As a result, repeated retransmission is not the problem in wired networks, but leads to waste the limited battery power of mobile device in MANETs.

In MANETs, packet losses is usually caused by high bit error rate. However, TCP can not distinguish between congestion loss and link error loss. Whenever a packet is lost, TCP assumes that network congestion has happened since it is designed for wired networks originally. It then invokes the congestion control such as reducing congestion window. When packet loss has occurred due to link error, TCP interprets the packet loss as congestion. In this case, TCP unnecessarily invokes congestion control. As a result, its performance suffers from the unnecessary congestion control, causing reduction in throughput and link utilization [3, 4].

2.2 TCP-Veno

The algorithm of TCP-Veno is mostly based on the algorithm of TCP-Vegas. In TCP-Vegas, the sender measures the $Expect\ Throughput$ and $Actual\ Throughput$

as described in Figure. 1, where $cwnd$ is the current congestion window size, $BaseRTT$ is the minimum of measured round-trip times, RTT is the smoothed round-trip time measured, and N is the backlog at the bottleneck queue. TCP-Vegas attempts to keep N to small range by adjusting the congestion window size. If N exceeds $\beta(=3)$, TCP-Vegas assumes that the network is congested [5].

$$Expect\ Throughput = cwnd\ /\ BaseRTT$$
$$Actual\ Throughput = cwnd\ /\ RTT$$
$$N = (Expect\ Throughput - Actual\ Throughput) \times BaseRTT$$

Fig. 1. TCP-Vegas Algorithm

TCP-Veno modifies the AIMD(Additive Increase Multiplicative Decrease) of TCP-Reno based on the algorithm of TCP-Vegas. When $cwnd$ exceeds $ssthresh$(slow-start threshold), TCP-Veno reduces the increase rate of $cwnd$ to avoid congestion as defined in Figure 2. If N is below β, TCP-Veno assumes that the available bandwidth is not fully utilized. Therefore, TCP-Veno increases the $cwnd$ by one after each round-trip time. If N exceeds β, TCP-Veno assumes that the available bandwidth is fully utilized. Therefore, TCP-Veno increases the $cwnd$ by one after two round-trip times [6].

```
if(cwnd ≥ ssthresh)
    if(N < β)
        cwnd = cwnd + 1
    else
        cwnd = cwnd + 1/2
```

Fig. 2. TCP-Veno Additive Increase Algorithm

Moreover, TCP-Veno can distinguish between the congestion loss and the link error loss. When the sender receives the 3 duplicative ACKs in AI(Additive Increase) phase, TCP-Veno adjusts $ssthresh$ as designed in Figure 3. If N exceeds β, TCP-Veno assumes that the packet loss has occurred due to congestion. Therefore, TCP-Veno adjusts the $ssthresh$ and $cwnd$ to a half of $cwnd$ for alleviating the congestion. If N is below β, TCP-Veno assumes that the packet loss has occurred due to link error. In this case, TCP-Veno reduces the $ssthresh$ and $cwnd$ by a smaller amount.

In this manner, TCP-Veno effectively avoids repeated packet losses, retransmissions, and the unnecessary congestion control by using the dynamic adjustment of congestion window. However, like TCP-Vegas, TCP-Veno can also save the problem of incorrect $BaseRTT$. The $BaseRTT$ is the smallest RTT without competitive flow in the network. Therefore, it is hard that the $BaseRTT$ is correctly estimated. Because of this problem, TCP-Veno can not monitor the network state. In addition, the low utilization of bandwidth and the inaccuracy of discrimination loss type are caused by the inaccurate $BaseRTT$ [7, 8].

if($N > \beta$)
    $cwnd = cwnd \ / \ 2$ //Congestion Loss
else
    $cwnd = cwnd \ / \ 0.8$ //Link Error Loss

Fig. 3. TCP-Veno Multiplicative Decrease Algorithm

3 TCP-New Veno

In this section, we propose TCP-New Veno for improving the performance of TCP-Veno and describe the monitoring algorithm and the congestion control of TCP-New Veno.

3.1 The Monitoring Algorithm of the Network State

TCP-New Veno adds the monitoring algorithm of the network state for solving the problem of $BaseRTT$. Our TCP-New Veno classifies the network state into three states, that is, stable, congestion increase, and congestion decrease.

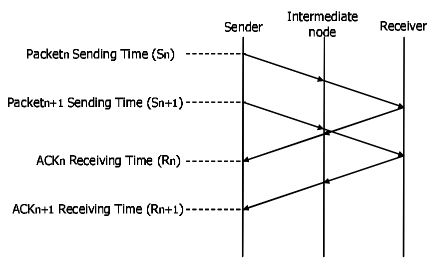


Fig. 4. Stable State of Network

Figure 4 shows the stable state of network. In this state, the sender sends $packet_n$ and $packet_{n+1}$ to receiver without queuing delay at S_n and S_{n+1} respectively. Then, the sender receives the ACKs for $packet_n$ and $packet_{n+1}$ at R_n and R_{n+1} respectively. Because of no queuing delay, the ACK receiving interval $R_{n+1} - R_n$ is equal to the packet sending interval $S_{n+1} - S_n$.

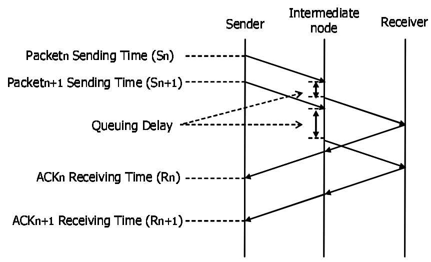


Fig. 5. Congestion Increase State of Network

Figure 5 shows the congestion increase state of network. In this state, the packet is queued due to the congestion increase in the queue of immediate node. Since the packet queuing delay is increased, the ACK receiving interval is increased. Therefore, the ACK receiving interval exceeds the packet sending interval.

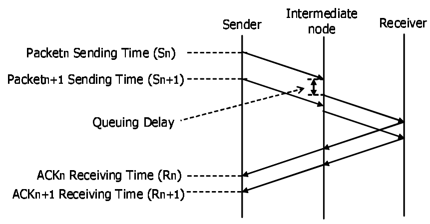


Fig. 6. Congestion Decrease State of Network

Figure 6 shows the congestion decrease state of network. Since the congestion decrease, the packet queuing delay is decreased. The ACK receiving interval is shorter than the packet sending interval.

$$\begin{aligned} Dq &= ACKReceivingInterval - PacketSendingInterval \\ &= (R_n - R_{n-1}) - (S_n - S_{n-1}) \end{aligned} \tag{1}$$

$$\Delta Dq = Dq_n - Dq_{n-1} \tag{2}$$

In order to monitor the network state, TCP-New Veno defines Dq(Relative Queuing Delay) and ΔDq as in (1) and (2). In the stable state of network, Dq is the zero because the ACK receiving interval is equal to the packet sending interval. In the congestion increase state of network, Dq has a positive value because the ACK receiving interval exceeds the packet sending interval. In the congestion decrease state of network, Dq has a negative value because the ACK receiving interval is below the packet sending interval. ΔDq, the difference of Dq, means the degree of the congestion increase rate. Figure 7 and Figure 8 show the summary of these concepts.

$Dq < 0$	//Congestion Decrease State
$Dq = 0$	//Stable State
$Dq > 0$	//Congestion Increase State

Fig. 7. Network State Classification Based on Dq

$\Delta Dq < 0$	//Congestion Degree Decrease State
$\Delta Dq = 0$	//Stable State
$\Delta Dq > 0$	//Congestion Degree Increase State

Fig. 8. Network State Classification Based on ΔDq

3.2 The Congestion Control

TCP-Veno estimates the network state by using N and β. TCP-Veno still inherits the $BaseRTT$ problem from TCP-Vegas. Thus, TCP-Veno can't accurately estimate the network state. To solve the problem, TCP-New Veno introduces the concept of Dq and ΔDq. By using these parameters, TCP-New Veno modifies the AIMD defined in TCP-Veno.

```
if (N < β)
    set cwnd = cwnd + 1
if (N ≥ β)
    if (Dq < 0)            // Highly Aggressive Phase
        set cwnd = cwnd + 1
    if (Dq = 0)            // Aggressive Phase
        if (ΔDq ≤ 0)
            set cwnd = cwnd + 1
        if (ΔDq > 0)
            set cwnd = cwnd + 1/2
    if (Dq > 0)            // Conservative Phase
        if (ΔDq ≤ 0)
            set cwnd = cwnd + 1/4
        if (ΔDq > 0)
            set cwnd = cwnd
```

Fig. 9. Additive Increase Algorithm in TCP-New Veno

Figure 9 shows the additive increase algorithm of TCP-New Veno. If N is below β, TCP-New Veno increases the *cwnd* by one after each round-trip time because the available bandwidth is not full. If N exceeds β, TCP-New Veno classifies the *cwnd* adjustment in three phases, that is, highly aggressive, aggressive, and conservative, based on the value of Dq. First, in the highly aggressive phase, network congestion is getting better. In this phase, TCP-New Veno increases the *cwnd* by one after each round-trip time. Second, the network state is stable in aggressive phase. Since TCP-Reno increases aggressively the *cwnd* in the stable state of network, TCP-New Veno also increases aggressively the *cwnd*, based on ΔDq, to compete with TCP-Reno. Third, the conservative phase means that the available bandwidth is nearly full. Therefore, if ΔDq is the negative, TCP-New Veno increases conservatively the *cwnd*. If ΔDq is the positive, TCP-New Veno keeps the current *cwnd*.

```
if (N ≥ β)
     if (Dq > 0)            // Congestion Loss
          set ssthreshold = cwnd / 2
          set cwnd = ssthreshold
else
     Keep all parameters
```

Fig. 10. Multiplicative Decrease Algorithm in TCP-New Veno

Figure 10 shows the multiplicative decrease algorithm of TCP-New Veno. In TCP-Veno, when the sender receives three duplicative ACKs, it tries to find the cause of the packet loss by using N and β. However, due to the problem of inaccurate $BaseRTT$, TCP-Veno can't accurately find the came out. To solve the problem in TCP-Veno, TCP-New Veno introduces the concept of Dq. If N exceeds β and Dq is the positive, TCP-New Veno assumes that the network is congested. Therefore, when packet loss is detected in this state, TCP-New Veno invokes the congestion control, like TCP-Reno.

TCP-New Veno modifies the AIMD employed in TCP-Veno by introducing new parameters, Dq and ΔDq. According to Dq and ΔDq, TCP-New Veno adjusts dynamically the *cwnd*. It can also discriminate the cause of packet loss. Therefore, TCP-New Veno can avoid repeating the packet loss and retransmission, and invoking the unnecessary congestion control. As the results, TCP-New Veno improves significantly the energy efficiency of mobile device and the bandwidth utilization.

4 Performance Evaluation of TCP-New Veno

In this section we evaluate the TCP-New Veno based on the ns-2 simulator [9]. This evaluation has been carried out to show some improvements on the throughput and the energy efficiency of TCP-New Veno in experimental networks. The performance of TCP-New Veno is compared with the performance of TCP-Reno and TCP-Veno.

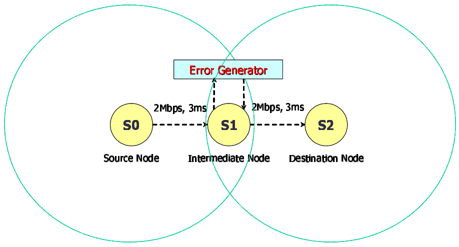


Fig. 11. Network Configuration for Simulations

4.1 Simulated Environment

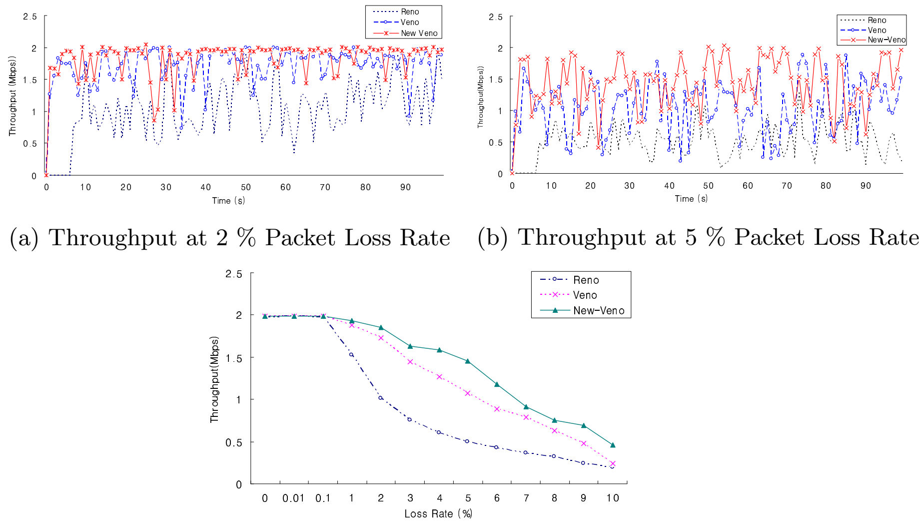
The network configuration for the simulation is shown in Figure 11. The simulation is based on mobile nodes according to IEEE 802.11 with 2Mbps and a nominal transmission radius of 250m on the link layer [10]. We used the Ad-hoc On-demand Distance Vector(AODV) routing protocol [11]. Since we focus on performance in presence of link errors, the mobility of mobile node is excluded in this simulation. To generate the link errors, the intermediate node(S1) drops compulsively the packets as the packet loss rate. The ranges of packet loss rate is 0∼10 % . The source node(S0) sends continuously packet to the destination node(S2). The initial energy of the mobile nodes set to 100J(Joule). All nodes consume the 0.6W(Watt) for transmitting a packet and 0.3W for receiving a packet.

4.2 Simulation Results

When packet losses have occurred due to the link error, Figure 12 shows the throughput of TCP-Reno, TCP-Veno and TCP-New Veno according to packet loss rate each. For 2% packet loss rate, Figure 12 (a) shows that TCP-New Veno performs better than TCP-Reno and nearly same as TCP-Veno. However, in Figure 12 (b) for 5% packet loss rate, TCP-New Veno performs much better than TCP-Reno and TCP-Veno. In Figure 12 (c) at 0∼10% packet loss rate, the performance of TCP-New Veno is better than TCP-Reno and TCP-Veno since TCP-New Veno can discriminate accurately the cause of packet loss and avoid effectively the unnecessary congestion control. As the packet loss rate increases higher than 5% the performance of TCP-New Veno is decreased as TCP-Reno and TCP-Veno. It is because that timeout is invoked due to the ACK loss.

$$\text{Energy Efficiency}(\eta) = \frac{Throughput}{Consumed\ Energy}(Kb/sJ) \qquad (3)$$

We also investigated the energy efficiency of each protocol in a lossy link situation. The energy efficiency of TCP-Reno, TCP-Veno and TCP-New Veno are shown in Figure 13. To evaluate the energy efficiency, we use the (3). It means the amount of bit that sender can send by a Joule. During the whole simulation



(a) Throughput at 2 % Packet Loss Rate (b) Throughput at 5 % Packet Loss Rate

(c) Average Throughput vs Packet Loss Rate

Fig. 12. Throughput vs Packet Loss Rate

time, 150sec, TCP-New Veno accomplishes better energy efficiency than TCP-Reno and TCP-Veno. The improvement is approximately about 20% $\sim$ 90% . At 5% packet loss rate, the throughput of TCP-Reno, TCP-Veno and TCP-New Veno are each of about 0.5Mbps, 1.3Mbps and 1.6Mbps. And the consumed energy of them is each of about 41J, 44J and 43J. Here, TCP-Reno has the low energy efficiency as the low throughput and high consumed energy. It means that TCP-Reno retransmits the large amount of packets. The consumed energy of TCP-New Veno is similar to TCP-Veno but the throughput of TCP-New Veno is higher than TCP-Veno. Therefore, TCP-New Veno has the better energy efficiency than TCP-Veno. This result shows that TCP-New Veno can avoid effectively the repeated retransmissions since it adjusts accordingly the *cwnd* by Dq and ΔDq.

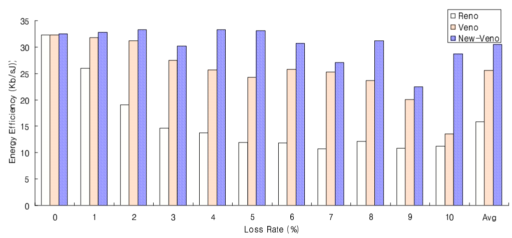


Fig. 13. Energy Efficiency VS Packet Loss Rate

5 Conclusion and Future Work

The performance of TCP degrades in MANETs since it can not distinguish between causes of packet losses. TCP does not also consider the energy of mobile node. In this paper, we propose the new transport protocol, called TCP-New Veno, to solve the problems of TCP in MANETs. The performance enhancement and and the energy efficiency in TCP-New Veno are obtained by avoiding the repeated retransmissions, by distinguishing between causes of packet loss, and by adjusting the *cwnd* to the current network state. The simulation results prove that TCP-New Veno has a better performance than TCP-Reno and TCP-Veno in MANETs.

In the future, we will focus our attention on the extension of TCP-New Veno such as mobility support and route change.

Acknowledgement

This research is supported by the MIC(Ministry of Information and Communication), Korea, under the ITRC(Information Technology Research Center) support program supervised by the IITA(Institute of Information Technology Assessment). This research also has been conducted by Korea Science and Engineering Foundation under contract number R01-2005-0000-10934-0.

References

1. The Internet Engineering Task Force: Mobile Ad-hoc Networks Discussion Archive-Data.
   http://www1.ietf.org/mail-archive/working-groups/manet/current/maillist.html
2. H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz: A Comparison of Mechanisms of Improving TCP Performance over Wireless Links. ACM Transactions On Networking. (1997)
3. H. Singh, S. Saena, and S. Singh: Energy Consumption of TCP in Ad Hoc Networks. Wireless Networks. (2004)
4. G. Holland and N. Vaidya: Analysis of TCP Performance over Mobile Ad Hoc Networks. ACM MOBICOM‘99. (1999)
5. L. Brakmo and L. Peterson: TCP Vegas: End-to-end Congestion Avoidance on a Global Internet. IEEE Journal on Selected Areas in Communications. (1995)
6. C. P. Fu and S. C. Liew: TCP Veno: TCP Enhancement for Transmission over Wireless Access Networks. IEEE Journal of Selected Areas in Communications. (2003)
7. T. Henderson and E. Sahouria: On Improving the Fairness of TCP Congestion Avoidance. IEEE GLOBECOM‘99. (1999)
8. W.Feng and S. Vanichpun: Enabling Compatibility Between TCP Reno and TCP Vegas. The Symposium on Applications and the Internet‘2003. (2003)
9. UCB LBNL VINT: Network Simulator(ns-2).
   http://www.isi.edu/nanam/ns/
10. C. E. Perkins and E. M. Royer: Ad-hoc On-Demand Distance Vector Routing. IEEE Workshop on Mobile Computing Systems and Applications. (1999)
11. M. Gast: 802.11 Wireless Networks: The Definitive Guide. O‘REILLY (2002)