Evaluation of a Queue Management Method for TCP Communications over Multi-hop Wireless Links

Satoshi Ohzahata and Konosuke Kawashima

Department of Computer, Information and Communication Sciences,
Tokyo University of Agriculture and Technology, Japan
e-mail: {ohzahata, k-kawa}@cc.tuat.ac.jp

***Abstract*: In this paper we propose a queue management method for improving TCP performance over multi-hop wireless links. With our proposed control method, packet loss in the wireless link is completely eliminated and all packets are delivered in the correct order. No change is required to TCP itself or to the servers. Our proposed method is implemented between the media access control layer (MAC) and logical link layer in each node, and is designed to help the local retransmission control in the MAC layer. Computer simulation experiments show that our proposed method can communicate in high error wireless link conditions even when conventional methods cannot maintain communication. The fairness problem of TCP communication, between connections with different round trip times, is also improved.**

1. Introduction

Wireless Internet environments have become widely used, since nowadays they can be constructed more easily and flexibly than wired Internet installations. In ubiquitous network environments, all devices, or at least many of them, are connected by multi-hop wireless links and controlled by Internet technology.

Reliable data communication in the Internet is mainly controlled by TCP[1],[2]. Since TCP was originally designed for wired networks, the control method assumes that packet loss generally occurs due to network congestion, and so when congestion occurs, it reduces the congestion window size of the traffic. The application of congestion control is, however, also applied when the packet loss occurs due to wireless link errors, and this always decreases the throughput of the connection. Furthermore, the problem becomes worse in multi-hop wireless networks.

To improve TCP performance in wireless environments, many methods have been proposed for adapting the congestion control to wireless environments. Selective Acknowledgment[3] notifies multiple lost packets by means of the acknowledgment (ACK). k-SACK[4] modifies the congestion detection and avoidance mechanisms of SACK with TCP New Reno for wireless environments. In Freeze-TCP[5], a receiver lets a sender keep the congestion window size over a frequently disconnected communication route without any modifications of the sender and the intermediate nodes. In I-TCP[6], the connection is split into two parts - wired networks and the wireless last hop. Packets lost in the wireless link are retransmitted locally to or from the splitting point of the base station (BS) without end-to-end control. Snoop agent[7] works at the Logical Link (LL) layer in a BS and monitors packet loss in the wireless link by checking the sequence number of the TCP header for each packet. If the agent detects a loss, the lost packet is retransmitted locally from the cache in the agent. In explicit loss notification (ELN)[8], the BS detects packet loss in the wireless link, and then explicitly notifies the fact to the sender. Based on the notification, the sender can select the most suitable congestion control depending on the reason of the loss. In the technique described in [9], ELN is introduced above the snoop mechanism, to notify the sender of the occurrence of a packet loss in the wireless link.

The above methods apply to one-hop (final hop) wireless networks, but there are additional problems in multi-hop mobile/wireless networks. Split TCP[10] extends the splitting scheme of I-TCP for mobile ad hoc networks to adapt to route changes resulting from mobility of the terminal. In TCP-BuS[11], every node can buffer during a route disconnection and reestablishment. In [12] a system is described in which notification of explicit route failure is provided by an enhanced inter-layer control mechanism in the intermediate node. The TCP agent can correctly control the congestion window size from this information.

These methods improve TCP performance in the last one-hop or in multi-hop wireless Internet environments, but also require complicated control and a lot of modifications to conventional systems.

We focus on TCP communication over lossy, multi-hop wireless links. Our method adopts a different approach to the above methods. We do not modify TCP itself nor the reference of sequence number information in TCP or LL headers, but packet losses are completely eliminated by enhanced retransmission control in the MAC layer, with a queue management method which maintains fairness for each flow. With this control, the packets are also delivered in the correct order even over the wireless link and the sender never receives a duplicate ACK from the receiver due to a wireless link error. The TCP agent of the sender can control the traffic to the wireless terminal as if the terminal were connected with a wired link. Above procedures are realized very simple mechanism.

To evaluate our proposed method, we executed computer simulation experiments. The results show that our method can maintain communication in high error wireless link conditions, even when communication is not possible with conventional TCP. The fairness problem of TCP communications between different round trip time connections is also improved.

The rest of this paper is structured as follows. Section 2 describes our proposed queue management method. In Section 3, the computer simulation experiments are de-

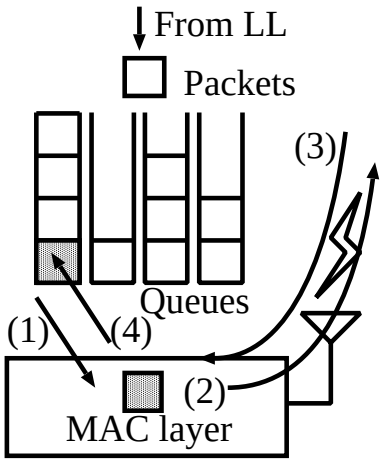

Figure 1: Steps in proposed queue management procedure in a node.

scribed to evaluate our method. Finally, Section 4 provides conclusions.

2. Basic Idea of Our Queuing Method

For reliable communication over a wireless link, most kinds of MAC layer provide a mechanism for local retransmission in the event of a wireless link error, but a maximum number of the retransmission attempts is defined to avoid certain packets wasting the bandwidth through excessive repeating of retransmissions. In our proposed method, the local retransmission mechanism is enhanced to execute the retransmission control indefinitely, to improve TCP performance. However, although this control is executed until each packet is received correctly at the other node, it is not permitted for a certain packet in a certain flow to occupy the bandwidth unduly, to the detriment of other flows.

Figure 1 shows our proposed method. A queue is provided between the MAC layer and LL layer for each node, because the radio condition differs at each node, and, if a common queue is used for all nodes, a packet to one node, which is suffering extremely bad radio conditions, may prevent the transmission of packets to the other nodes.

When one of the queues is allowed to send a packet, (1) the head packet of the queue is copied to the MAC layer for transmission, but the original continues to stay at the head of the queue. In the next step, (2) the MAC layer forwards the packet to the receiver. If the transmission is successful, (3) the acknowledgment is notified from the MAC layer in the receiver, and (4) this is also signaled to the queue, to delete the packet from the head of queue. Otherwise, the MAC layer continues retransmitting the packet, until the acknowledgment is received. In case that the packet is successfully transmitted, the next queue is arrowed to transmit the head of packet.

If local retransmission cannot result in recovery of the packet loss, i.e., the acknowledgment in Figure 1 (3) cannot be received, the period of time attempting retransmission will exceed the predefined limit. In this case, the packet is eliminated from the MAC layer, and the next queue is allowed to send a packet. Since the eliminated packet remains at the head of the queue, it can be retransmitted to the receiver again when the queue has its next turn to send.

Figure 2: Simulation model.

Note that the sequence numbers in the TCP and LL headers are not referred to in the above procedures. It should also be emphasized that the packet order is never changed as a result of a wireless link error in our proposed method. Thus, even when there is a high error rate on the wireless link, the TCP agent in the sender recognizes that the wireless link has a narrow bandwidth or requires a long round trip time. In such a case, the queue overflows and some packets are lost. However, since the TCP congestion control quickly adjusts the flow, overflow of the queue seldom occurs. In addition, since all ACK packets are delivered to the sender without fail, the TCP retransmission timer hardly ever times out or, if it does so, it is because the wireless link conditions are so poor that it is impracticable to continue the TCP connection. To confirm these principles, the next section describes the evaluation of our proposed method.

3. Simulation Experiments

To evaluate our proposed queue management method, computer simulation experiments were executed using the network simulator ns-2.1b8[13]. We compared two queue management methods: the original Deficit Round Robin (DRR)[14] method and our proposed method, which is also implemented on DRR. In both methods, the queue management procedure uses a separate queue for each node. We compare these two queue management methods in TCP communications using the network model shown in Figure 2. Common conditions for all simulation experiments are as follows.

[Queue conditions]

- In DRR, a queue is prepared for a flow (destination address). All the queues are constructed with a shared memory, and the total memory size to store packets is 50 packets.

[Network conditions]

- The transfer speed of each wireless link in Figure 2 is 2Mb/s. These nodes form a string topology of nodes, and the distance between each pair of nodes was fixed at 200m. The number of nodes, $n+1$, was varied from 2 to 8. These nodes do not move and so link failure never occurs as a result of mobility. Node 0 cannot communicate with node 2 because of the distance.
- For wireless communications, the each link error rate is selected from 5×10^{-5} or 1×10^{-6} error/bit for all links between nodes.

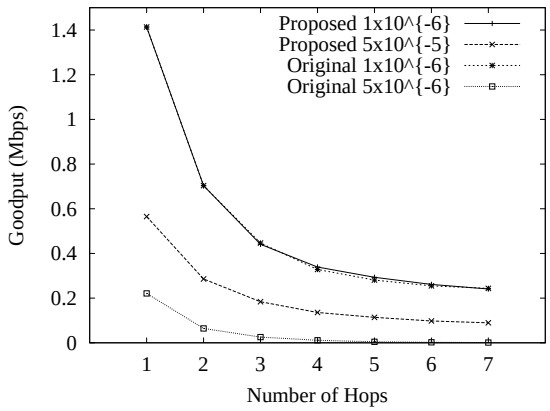

Figure 3: Average goodput at TCP level.

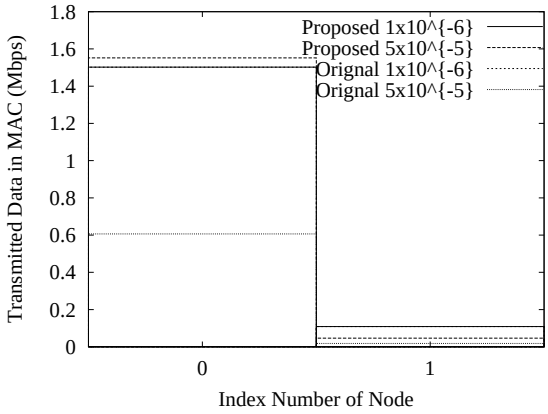

Figure 4: Average transmitted data in MAC level (One hop model).

- Medium access control is distributed foundation control (DFC) of IEEE 802.11 implemented in ns[13].
- The ad-hoc routing protocol used is destination sequenced distance vector (DSDV)[15].

[Traffic conditions]

- New Reno version of TCP.
- The source at the sender sends packets continuously to the receiving node using FTP under the TCP congestion control. The sizes of all TCP segments and ACKs are 1460 octets and 40 octets, respectively.
- The congestion window size in TCP is 40 segments, because the packet size is fixed in the simulation experiments.

3.1. Single-flow model

First, we experimented with changing the number of nodes from 2 to 8 under wireless link bit error rate conditions of 1×10^{-6} and 5×10^{-5}. In each case, the TCP source is at node 0 and the sink is at node n, and only a single flow is considered. The simulation time is 100000s for each experiment.

Figure 3 shows the average goodput (effective throughput)when the bit error rate of the wireless link is 1×10^{-6} and 5×10^{-5}, respectively. In these graphs, "Proposed 1x10^{-6}" means our method with a bit error rate of 1×10^{-6}, and "Original 1x10^{-6}" is the same as "Proposed 1x10^{-6}" except for the queue management method, which is that of the original DRR. The x-axis is the number of hops and the y-axis means goodput of at the TCP level.

In case of a bit error rate 1×10^{-6}, the two methods give almost identical results, for all the number of hops considered. With both methods, the goodput reduces as the number of hops increases, in spite of the fact that each wireless link error packet is recovered by the retransmission control of MAC layer. This is because the exposed terminal problem mentioned earlier reduces the goodput of multi-hop wireless environments. For example, in the case that node 2 is transmitting a data frame to node 3, the two adjacent nodes, node 1 and node 3, cannot send any frame and node 1 cannot receive any frame even if node 0 has a frame ready to send to node 1. The MAC control method cannot use the bandwidth effectively.

With a bit error rate of 5×10^{-5}, the original DRR method cannot maintain communication for more than a few hops because the retransmission mechanism of the MAC layer cannot cover the packet loss. Our proposed method, however, can maintain communication because the MAC can continue to retransmit a lost packet until the packet is received correctly at the next node. This control is executed repeatedly and so the frame is delivered correctly from the source node to the end node.

We also observed the TCP instability problem[16] for both methods when the number of nodes was from 3 to 8 in these simulation experiments. The traffic was stopped intermittently although there is no route failure or link breakage. This problem is due to the burst nature of TCP traffic, and the exposed terminal problem and contention in the IEEE 802.11 MAC. By reducing the congestion window size of TCP, this problem may be improved[16], however the goodput is also reduced in case of long round-trip time between the sender and receiver.

Figures 4 and 5 show the transmitted data in the MAC layer for one hop and seven hops, respectively. The throughput means how much bandwidth is used to transmit the data frames, since the control frames are not included in the throughput. We can see from these graphs how effectively the MAC bandwidth is used.

The x-axis is the index number of each node and the y-axis is the data transmitted from that node. Figure 4 shows the one hop case, in which node 0 sends TCP data, and node 1 receives it and sends the ACKs back to node 0. These two nodes use a single wireless link. With a bit error rate of 1×10^{-6}, the two methods gave the same result. With an error rate of 5×10^{-5}, our proposed method uses more bandwidth than in the case of an error rate of 1×10^{-6}, because the control method repeatedly retransmits the lost frames until they are delivered. Our method can consume the total bandwidth in the case of

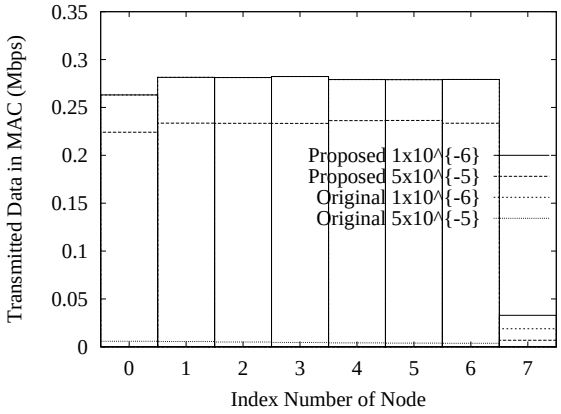

Figure 5: Average transmitted data in MAC level (seven-hop model).

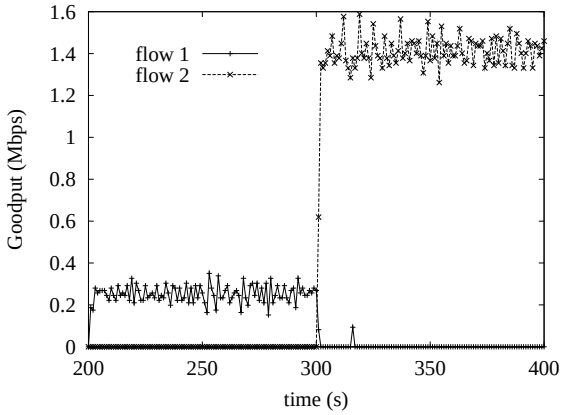

Figure 6: Goodput of flow 1 and flow 2 in DRR (1×10^{-6} bit error rate).

a high bit error rate. In fact, we observed that the queue length is never 0 at the node 0, which means that the maximum bandwidth of the wireless link is always used.

In fact, in case that the wireless link is single hop, our queue management method improves 30% of the throughput from the snoop[7] method at the maximum[17].

Figure 5 shows the seven hop case, in which node 0 sends TCP data, and node 7 receives it and sends the ACKs to node 0. With a bit error rate is 1×10^{-6}, both methods also gave the same results. In the case of a bit error rate of 5×10^{-5}, our proposed method cannot use the bandwidth as effectively as it could with an error rate of 1×10^{-6}. This is because the round trip time is increased from 290ms of 1×10^{-6} to 645ms of 5×10^{-5} and the throughput is reduced by the congestion control of TCP. The increment of round trip time is caused with increasing the retransmission for error frames and the contention in MAC layer. This increment of round trip time is, however, unavoidable because TCP connections are and the wireless LAN has a simplex link, and the TCP agent should receive the ACK from the receiver to send the next data to the receiver. In these conditions, the data flow of TCP and the ACK flow of TCP scramble for the simplex link and many contentions error occur. Furthermore, if the TCP agent sends many data packets, the same number of the ACK packets are returned. This also increases the contention errors.

3.2. Two flow model

In the next set of simulations, We prepare eight nodes in the network topology. We set up two TCP connections, flow 1 from node 0 to node 7 and flow 2 from node 3 to node 4. The bit error rate of the wireless links was set at 1×10^{-6} or 5×10^{-5}. Flow 1 started communication at 200s and flow 2 started at 300s and the both flows stopped at 400s. Figures 6–9 show the goodput of the two flows. In these results, the goodput was sampled every second.

Figure 6 shows the results for the original DRR method with a bit error rate is 1×10^{-6}. From 200s to 300s , flow 1 was transmitted with a performance the same as that shown in Figure 3. At 300s, flow 2 starts

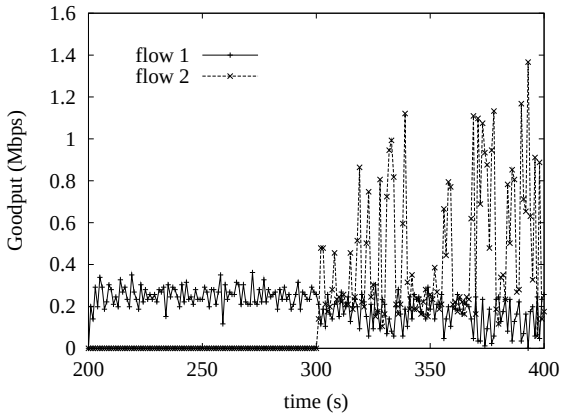

Figure 7: Goodput of frow 1 and flow 2 in our proposed method (1×10^{-6} bit error rate).

transmission, and no bandwidth is shared with flow 1 in spite of the fact that every node uses DRR as the queue management method. This problem is mentioned in Ref. [16] as the unfairness problem between the two flows. In this situation, every TCP segment of flow 1 is dropped at the wireless link between node 3 and node 4 due to contention in the MAC layer after the commencement of flow 2. This phenomenon interferes with the end-to-end TCP congestion control for flows which have started earlier but have a longer round trip time communication time, and communication disruption occurs.

Figure 7 shows the same results for our proposed method. For flow 1, the average goodput from 200s to 300s, and from 300s to 400s, are 0.253Mbps and 0.165Mbps, respectively. For flow 2, the average goodput from 300s to 400 is 0.445Mbps. From these results, it can be seen that the fairness problem is improved, because no TCP segment loss occurs in the wireless link. The bandwidth is, however, not shared fairly between flow 1 and flow 2. This is because flow 2 of the TCP flow control and the MAC allow burst traffic, and the round trip time of the flow 2 becomes temporarily long.

Figures 8 and 9 show the results of the two-flow model with a bit error rate of 5×10^{-5}. In the original DRR

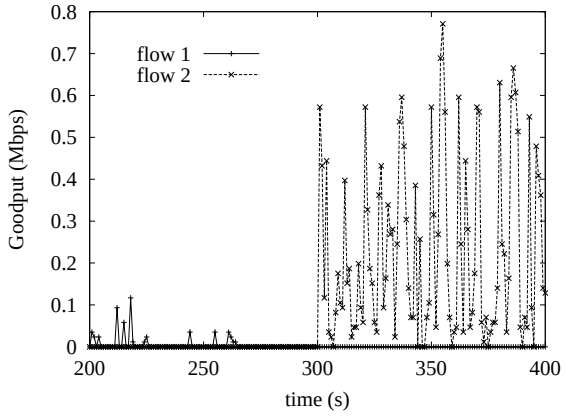

Figure 8: Goodput of flow 1 and flow 2 in DRR (5×10^{-5} bit error rate).

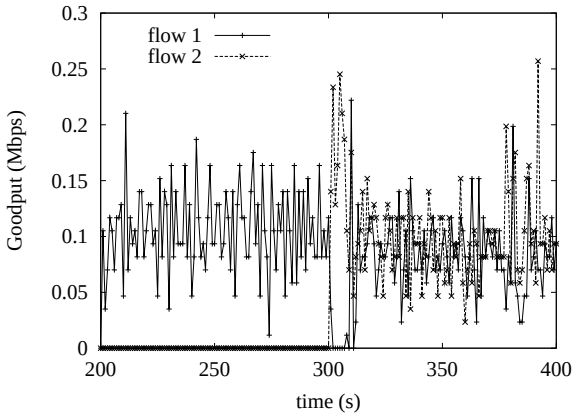

Figure 9: Goodput of flow 1 and flow 2 in our proposed method (5×10^{-5} bit error rate).

method, flow 1 cannot communicate as in Figure 3, and only the flow 2 achieves communication (Figure 8). In Figure 9, our proposed method improves the unfairness problem between the two flows with different round-trip times. The end-to-end flow control works well, since these data segments and the ACK segments are transferred correctly even under high bit error rate conditions. For flow 1, the average goodput from 200s to 300s, and from 300s to 400s, are 0.107Mbps and 0.079Mbps, respectively. For flow 2, the average goodput from 300s to 400s is 0.104Mbps. However, flow 1 was disconnected when flow 2 started communication, because the slow start control results in burst traffic and the segments of the flow 1 had to wait a long time until their turn for transmission. In high bit error rates, the bandwidth of the wireless link is consumed by repeated retransmission of the corrupted packets, and so the goodput of the MAC media is reduced. This makes the problems worse, however the phenomenon is immediately improved with congestion control of the two TCP agent.

3.3. Four flow model

In the next set of simulations, we set up four TCP connections, flow 1 is from node 0 to node 7, flow 2 is from node 6 to node 1, flow 3 is from node 2 to node 5 and flow 4 is from node 4 to node 3. Flow 1 started communication at 200s, flow 2 started at 300s, flow 3 started at 400s and flow 4 started at 500s. Flow 1 – 4 stopped at 600s. Bit error rate of all the wireless links was set at 1×10^{-6} or 5×10^{-5} in each experiment. Figures 10–13 show the goodput of the four flows. In these results, the goodput was also sampled every second.

Table 1: Average goodput of flow 1 – flow 4 in DRR method in 1×10^{-6} bit error rate (Mbps).

seconds	200–300	300–400	400–500	500–600
flow 1	0.247	0.098	0.003	0
flow 2	-	0.177	0.011	0.002
flow 3	-	-	0.414	0.085
flow 4	-	-	-	1.137
All	0.247	0.276	0.428	1.22

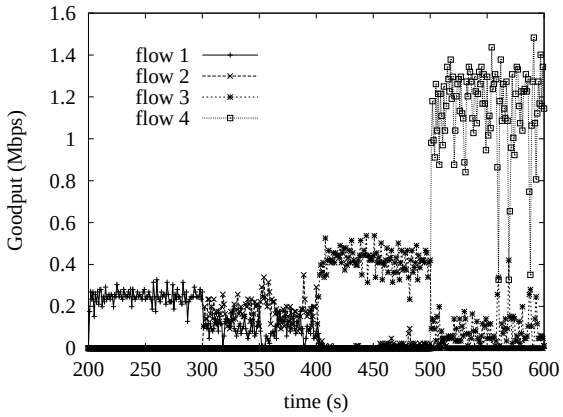

Figure 10: Goodput of flow 1 – flow 4 in DRR method (1×10^{-6} bit error rate).

Figures 10, 11 and Tables 1, 2 show the results of the four-flow model with a bit error rate of 1×10^{-6}.

As shown in Figure 10 and Table 1, the orignal DRR method cannot fairly share the bandwidth. In 500–600s, flow 1 and flow 2 cannot keep the communications because the shorter round trip time flows, flow 3 and flow 4, extremely consume the bandwidth with the congestion control of TCP. We can say that it is difficult to maintain multiple FTP communications over TCP congestion controls in multi-hop wireless LAN networks even if there is no packet loss in the wireless links.

Our proposed method, however, the four flows can keep the connections for the simulation periods in Figure 11 and Table 2. Between flow 1 and flow 2, each bandwidth is fairy shared even if two, three and four flows are contained in the networks. For 400-500s and 500-600s, the flow 3 or flow 4 gain larger bandwidth than that of the other flows. This is because that the flow 3 and flow 4 have a smaller round trip time than those of the flow 1 and flow 2. This phenomenon is unavoidable in conventional congestion controls of TCP. However, in case that we focus on the flow 1, the shared bandwidth is almost

Table 2: Average goodput of flow 1 – flow 4 in our proposed method in 1×10^{-6} bit error rate (Mbps).

seconds	200–300	300–400	400–500	500–600
flow 1	0.247	0.139	0.0732	0.0571
flow 2	-	0.143	0.0802	0.0588
flow 3	-	-	0.214	0.0787
flow 4	-	-	-	0.587
All	0.247	0.282	0.367	0.666

Table 3: Average goodput of flow 1 – flow 4 in DRR method in 5×10^{-5} bit error rate (Mbps).

seconds	200–300	300–400	400–500	500–600
flow 1	0.001	0.0035	0.0002	0.0004
flow 2	-	0.0079	0.0027	0
flow 3	-	-	0.0366	0.0263
flow 4	-	-	-	0.1666
All	0.001	0.0114	0.0395	0.1933

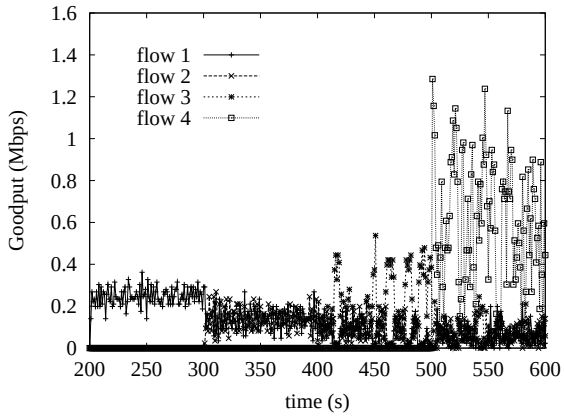

Figure 11: Goodput of flow 1 – flow 4 in our proposed method (1×10^{-6} bit error rate).

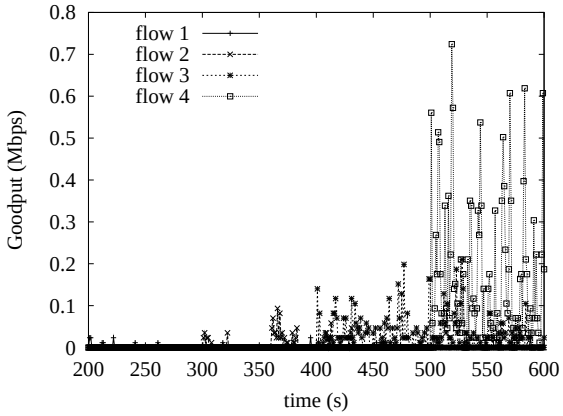

Figure 12: Goodput of flow 1 – flow 4 in DRR method (5×10^{-5} bit error rate).

same as 0.247/(the number of flows) Mbps in each period of 100s. The unfairness problem is improved in the point.

Figures 12, 13 and Tables 3, 4 show the results of the four-flow model with bit error rate of 5×10^{-5}. Figure 12 and Table 3 shows the results of the original DRR. The flow 1 and flow 2 cannot keep their communications in the all period and the flow 3 and flow 4 only keep their narrow bandwidth.

Figure 13 and Table 4 show the results of our proposed method with error prone link conditions, and the bandwidth unfairness problem is improved from no error conditions in the results. This is because that local retransmission control in MAC layer is required for the error frame and the latter frames should stay at the queue of the node until the error frame is delivered to the receiver correctly. This control reduces a burst nature of TCP congestion control for shorter round trip time flows. These results show that our proposed method works well in multiple TCP flow conditions over error prone links.

4. Conclusions

We have evaluated our proposed queue management method in lossy multi-hop wireless environments. Since the proposed method is designed to help local retransmission in the MAC layer, packet loss in a wireless link is completely eliminated and all packets are delivered in the correct order. Moreover, the proposed method is simple to implement in a conventional system. These simulation experiments show that our method is effective for such environments.

However, our method still suffers from the burst nature of TCP traffic, and the burst control and contention problems of IEEE 802.11 media access control, and we need to solve these problems. In error prone link conditions, TCP connections have long round trip time, and this problem is unavoidable to correctly deliver segments and keep the connections. This may cause time out of the congestion timer of TCP and we should also improve the problems. Our method is currently only applicable to fixed wireless multi-hop environments, and we need to improve it for mobile ad-hoc communication environments. The traffic model used in our simulations is also simple and we also need to refine it to model multimedia traffic.

Further sophisticated control for multimedia traffic should be considered for the proposed method. For example, the addition of Random Early Discard, Explicit Congestion Notification or Class Based Queuing will improve the performance for realistic network traffic.

REFERENCES

[1] M. Allman, V. Paxson and W. Stevens, *"TCP Congestion Control,"* RFC2581, 1999.

[2] S. Floyd and T. Henderson, *"The New Reno Modification to TCP's FastRecovery Algorithm,"* RFC2582, 1999.

[3] M. Mathis, J. Mahdavi, S. Floyd and A. Romanow, *"Selective Acknowledgment Options,"* RFC2018,

Table 4: Average goodput of flow 1 – flow 4 in our proposed method in 5×10^{-5} bit error rate (Mbps).

seconds	200–300	300–400	400–500	500–600
flow 1	0.111	0.0663	0.0462	0.0323
flow 2	-	0.0531	0.0466	0.0471
flow 3	-	-	0.0491	0.0466
flow 4	-	-	-	0.0539
All	0.111	0.119	0.142	0.180

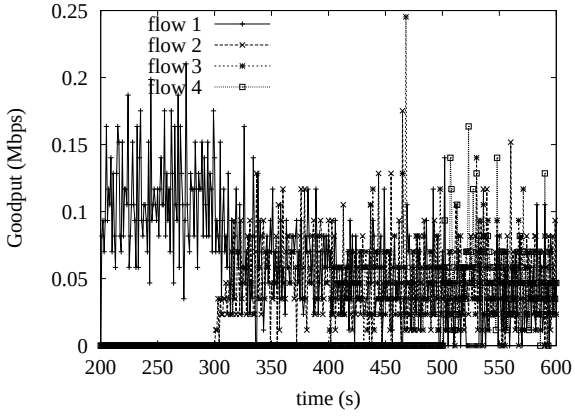

Figure 13: Goodput of flow 1 – flow 4 in our proposed method (5×10^{-5} bit error rate).

1996.

[4] A. Chrungoo, V. Gupta, H. Saran and R. Shorey, *"TCP k-SACK: A Simple Protocol to Improve Performance over Lossy Links,"* Proc. IEEE Globecom'02, pp. 1713–1717, 2001.

[5] T. Goff, J. Moronski, D. S. Phatak and V. Gupta, *"Freeze-TCP: A true end-to-end TCP enhancement mechanism for mobile environments,"* Proc. IEEE Infocom'00, pp. 1537–1545, 2000.

[6] A. Bakre and B. R. Badrinath, *"I-TCP, Indirect TCP For Mobile Host,"* in 15th International Conference on Distributed Computing System, 1995.

[7] H. Balakrishnan S. Seshan, and R. H. Katz, *"Improving reliable transport and handoff protocols in cellular wireless networks,"* ACM Wireless Networks, Vol. 1, Issue 4, pp. 469–481, 1996.

[8] H. Balakrishnan and R. H. Katz, *"Explicit loss notification and wireless web performance,"* IEEE Globecom'98, http://nms.lcs.mit.edu/papers/globecom98/, 1998.

[9] W. Ding and A. Jamalipour, *"Delay performance of the new explicit loss notification TCP technique for wireless networks,"* IEEE Globecom'01, pp. 3483–3487, 2001.

[10] S. Kopparty, S. V. Krishnamurthy, M. Paloutsos and S. K. Tripathi, *"Split TCP for Mobile Ad Hoc Networks,"* IEEE Globecom'02, pp. 139–143, 2002.

[11] D. Kim, C.-K. Toh, and Y. Choi, *"TCP-BuS : Improving TCP Performance in Wireless Ad Hoc Networks,"* IEEE ICC'00, no. 1, pp. 1707–1713, 2000.

[12] D. Sun and H. Man, *"ENIC– An Improved Reliable Transport Scheme for Mobile Ad Hoc Networks,"* IEEE Globecom'01, no. 1, pp. 2852–2856, 2001.

[13] "Network Simulator ns (version2)," *http://www.isi.edu/nsnam/ns/.*

[14] M. Shreedhar and G. Varghese, *"Efficient Fair Queuing Deficit Round Robin,"* ACM Sigcomm'95, pp. 231–242, 1995.

[15] C. E. Perkins and P. Bhagwat, *"Highly Dynamic Destination–Sequenced Distance–Vector Routing,"* ACM Sigcomm'94, pp. 234–244, 1994.

[16] S. Xu and T. Saadawi, *"Does the IEEE 802.11 MAC Protocol Work Well in Multihop Wireless Ad hoc Networks ?,"* IEEE Comm. Mag., June, pp. 130–137, 2001.

[17] S. Ohzahata, S. Kimura, Y. Ebihara and K. Kawashima, *"A Queue Management Method for Improving TCP Performance in Wireless Environments,"* IEEE WCNC'04, 2004. (to appear)