Rate-Based Transport Control for Mobile Ad Hoc Networks

Hongqiang Zhai, Xiang Chen and Yuguang Fang
Department of Electrical & Computer Engineering
University of Florida, Gainesville, Florida 32611-6130
Tel: (352) 846-3043, Fax: (352) 392-0044
E-mail: zhai@ecel.ufl.edu, xchen@ecel.ufl.edu, and fang@ece.ufl.edu

***Abstract*— The traditional congestion control mechanism, i.e., TCP, encounters a number of new challenges when applied in mobile ad hoc networks (MANET), such as wireless link error, medium contention, and frequent route failures. And very poor performance of TCP in MANET has been reported in many recent studies.**

In this paper, we focus on the problems resulting from the medium contention and propose a novel *R*ate *B*ased end-to-end *C*ongestion *C*ontrol scheme (RBCC). We first illustrate that, under the impact of medium contention, window based congestion control algorithm is unstable and hence may not be appropriate for MANET because the optimum congestion window size is very small and may be even less than one, i.e., the source should send less than one packet in one round trip time (RTT). Based on the novel use of channel busyness ratio, which, we show, is an accurate sign of the network utilization and congestion status, a new rate control scheme has been proposed to efficiently and reliably support the transport service in MANET. In RBCC, a sublayer consisting of a leaky bucket is added under TCP to control the sending rate based on the network layer feedback at the bottleneck node. Extensive simulations show that our scheme significantly outperforms traditional TCP in terms of channel utilization, delay, and fairness.

I. Introduction

Mobile ad hoc networks (MANETs) have found many applications in battlefield, disaster rescue and conventions, where fixed communications infrastructures are not available and quick network configurations are needed. To provide reliable transport service over and hence fully exploit the potential of MANETs, efficient congestion control is of paramount importance.

Unfortunately, traditional TCP congestion control mechanism performs very poorly, as shown in recent studies ([1]–[7] and reference therein). TCP congestion control has an implicit assumption, which is that any packet loss is due to network congestion. However, this assumption is no longer valid in the MANET as packet losses may well be due to channel bit errors, medium contention, and route failures.

To alleviate the impact of mobility, several schemes, such as those in [8]–[11], were proposed to distinguish between route failures from topology changes and network congestion through explicit route failure notifications. Other schemes, like those in [12], [13], do not use the network layer feedback but keep the TCP states unchanged when the source first detects the out-of-order packet and retransmission timeout.

Several works have already noticed that greedy TCP can result in severe congestion in MANET and hence suffer performance degradation. Link-RED in [4] were proposed to mark or drop TCP packet according to observed packet collisions to notify the TCP source to reduce congestion window size before it becomes excessive large. [1] dynamically adjusted the congestion window limit according to path length of TCP flows. In [14], a neighborhood RED scheme was proposed to address TCP fairness problem resulted from medium contention.

In this paper, we mainly focus on the problems arising from medium contention and show that a rate based congestion control protocol in stead of window based congestion control protocol should be more appropriate for MANET. In Section II, we illustrate that the optimum congestion window size of TCP may be less than one even in very simple topology, say chain topology, to maximize the end-to-end throughput and minimize the end-to-end delay. Therefore unstable performance is observed for TCP for large variation of sending rate.

There is already one equation-based rate control scheme for Internet, i.e., TFRC [23]. However, the recent studies in [22] shows that TFRC is very conservative in MANETs due to inaccuracy of loss prediction.

To conduct accurate end-to-end rate control, each node is in dire need of a robust and easy measured metric to control the feedback of each passing packet. For Internet and ATM network, lots of work have discussed the explicit and precise congestion feedback for end-to-end control, such as [15], [16]. These work are based upon the measure of packet loss, queue length, and link utilization, which are significantly different in MANETs. Severe medium contention is coupled with packet losses and large queue length when congestion happens in MANETs. And wireless link is shared by the neighboring nodes and is no longer a dedicated link between two nodes as in wired networks. Thus in Section II-B we propose to use channel busyness ratio as the timely and accurate sign of the network utilization as well as congestion.

Then in Section III we propose a novel rate based congestion control protocol (RBCC) based upon the novel use of channel busyness ratio. Each forwarding node allocate the channel resource to the passing flows by monitoring and mod-

0-7803-8966-2/05/$20.00 (C) 2005 IEEE

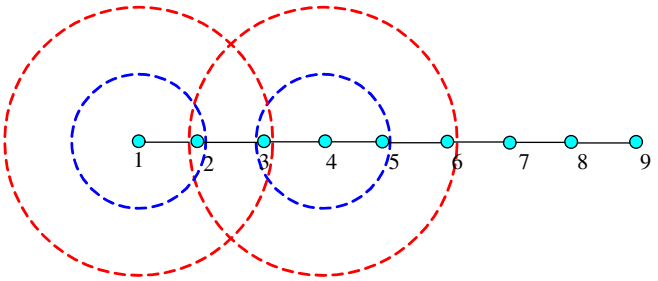

Fig. 1. Chain topology with 9 nodes. Small circles denote the transmission range, and the large circles denote the sensing range

ifying the feedback field of the data packets according to its measured channel busyness ratio. After receiving the feedback in the acknowledgement packets from the destination, the source accordingly adjust the sending rate. Thus, in RBCC, the sending rate of each flow is determined by the channel utilization status at the bottleneck node.

In Section IV, we evaluate the performance of the scheme through extensive simulations. Finally, conclusions and future work are given in Section V.

II. Medium Contention and Its Impact

A. Optimal Congestion Window Size for TCP and Ideal Sending Rate

We will show in this subsection that window based congestion control may not be an appropriate choice to support stable and reliable transport service in MANET because the optimal sending rate per RTT is very small and may be even less than one packet/RTT. Hence window based congestion control must results in instability.

The authors of [19] have shown that, in chain topology like that of Fig. 1, the maximum utilization of a chain of ad hoc nodes is $1/4$ by scheduling the nodes four hops away to simultaneously transmit. Thus the optimal sending rate R_o from the source should be just enough to make the above schedule feasible. Higher sending rate will result in packet collisions and losses, and hence long delay. At rate of R_o, the packet will be delivered to the destination in shortest time without any medium collision and queueing dealy, and RTT will be fixed and is denoted by RTT_o here. Assuming there are N TCP flows from node 1 to node 9, the optimal sending rate of each TCP flow is

$$R_{oeachtcp} = R_o/N, \tag{1}$$

$$K_{eachtcp} = RTT_o \times R_o/N, \tag{2}$$

where $K_{eachtcp}$ is the number of packets sent by each TCP source per RTT_o.

We use simulation to illustrate R_o when the 802.11 MAC is used. The channel bandwidth is 2 Mbps. A CBR/UDP flow with the same packet length as that of TCP DATA packets, i.e., 1000 bytes, starts from node 1 and destines for node 9. Another CBR/UDP flow with the same packet length of TCP ACK packets starts from node 9 and destines for node 1. Two UDP flows have the same sending rate. We gradually increase the sending rate until there are some packet losses for collisions in the 300 seconds simulation. The simulation results are summarized in Table I where the performance are also included when there are six TCP flows for comparison.

TABLE I

Simulation Results for TCP and UDP Flows

Traffic type	UDP (node 1 to 9)	6 TCP flows
Aggregate throughput (Kbps)	212	205
Average end-to-end delay (s)	0.0687	0.627
RTT(s)	0.132*	1.120
Dropped packets / s	0	1.276
$*$ the sum of average end-to-end delay of the two UDP flows		

Thus the optimum aggregate sending rate is 212 $Kbps \approx$ 25 pkt/s given that there is no packet loss due to collisions. And $RTT_o = 0.132s$ and $K_{eachtcp} = 0.55pkt/RTT$ when N = 6. The more TCP flows there are, the smaller $K_{eachtcp}$ is.

In summary, to provide high throughput, short delay and stable performance with few packet collisions, window-based congestion control is not appropriate for MANET. And we will design an efficient rate-based congestion control algorithm in the following sections.

B. Channel Busyness Ratio

We propose to use an easily measured metric *channel busyness ratio* r_b to track the current channel and network utilization at the location of each node. Notice that the IEEE 802.11 is a CSMA-based MAC protocol, working on the physical and virtual carrier sensing mechanisms. The channel is determined busy when the measuring node is sending, receiving, or its network allocation vector (NAV) [17] indicates the channel is busy, and to be idle otherwise.

In our previous work [24], [25], We have shown through both theoretical and simulation studies that the IEEE 802.11 DCF protocol could achieve the maximal channel utilization when it is working at the optimal point corresponding to a certain amount of arriving traffic. If the arriving traffic is heavier than this threshold, the network enters into saturation, resulting in significant increase in delay and decrease in throughput due to severe collisions; on the other hand, if the arriving traffic is less than this threshold, there is almost no collisions and channel capacity is wasted.

Further, if we denote by th_b the channel utilization corresponding to the optimal point, the available normalized throughput is proportional to $th_b - r_b$ and r_b is an accurate estimate of current channel utilization before the network achieves the maximal throughput. As shown in [24], th_b is almost the same for different number of active nodes and packet size, and $th_b \approx 0.95$ (with RTS/CTS). In this paper we set th_b to 92% accordingly and leave 3% space to avoid entering into saturation.

III. Rate-Based Congestion Control

Our proposed rate-based end-to-end control protocol (RBCC) is aimed to work around the optimal point in the sense

that the channel capacity is fully utilized while no congestion is caused. As mentioned earlier, in mobile ad hoc networks, the optimal window size for TCP is very small, typically less than 5 in packets and even less than 1. Hence any change in congestion window may result in large throughput oscillation in each RTT, failing to stabilize the throughput. For this reason, RBCC, unlike TCP, employs rate based control, in order to stabilize the throughput of each flow while making full use of the available bandwidth.

A. Protocol Overview

RBCC controls the sending rate of each flow by the explicit feedback carried in the ACKs. And each RBCC packet carries a congestion header as shown in Table II, which is used to communicate a flow's state to the intermediate nodes and the feedback from the intermediate nodes further to the destination. The field r_p is the sender's current permit arriving rate, and the field ci is the sender's currently used control interval. They are filled in by the sender and never modified in transit. The last field, fb, is initiated by the sender and all the intermediate nodes along the path may modify it to directly control the packet sending rate of the sources.

The RBCC sender maintains an estimate of the round trip time rtt and accordingly calculates the control interval ci. It also adjusts the permit arriving rate r_p according to the explicit feedback in the ACKs. Each time the sender transmits a packet, it attaches a congestion header to the packet with the latest r_p and ci.

All the nodes along the flow's path, including the RBCC sender and receiver, keep monitoring the channel busyness ratio r_b, and calculate the feedback accordingly. Then, according to the rules specified in 5.3, the node will decide whether and how to modify the fb field in the congestion header. The more congested node later in the path can overwrite the fb field in the congestion header. Ultimately, the packet will contain the feedback from the bottleneck node along the path. When the feedback reaches the receiver, it is returned to the sender in an ACK packet, and the sender updates its permit arriving rate r_p accordingly. The updated arriving rate r_p is then used by the sender to control the permit arriving rate at the leaky bucket.

TABLE II
CONGESTION HEADER

r_p (sender's permit arriving rate)
ci (sender's control interval)
fb (feedback)

A RBCC receiver is similar to a TCP receiver except that when acknowledging a packet, it copies the congestion header from the data packet to its ACK.

B. Rate Control Mechanism

The rate control mechanism of RBCC is illustrated in Fig. 2. A leaky bucket is attached to the transport layer to control the sending rate of RBCC sender. The permit arriving rate of the leaky bucket is dynamically adjusted according to the explicit

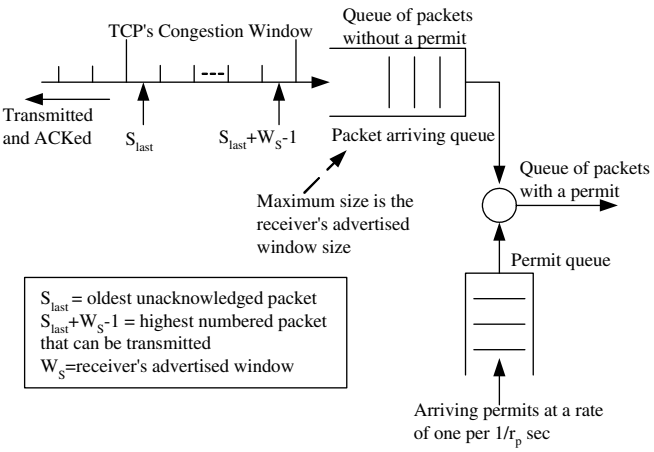

Fig. 2. Rate control mechanism

feedback carried in the returned ACK. A RBCC sender always sets its congestion window $cwnd$ to the receiver's advertised window size in packets.

Initially, when the RBCC sender sends out the first packet of a flow, $r_p = 0$, and ci is set to 0 to indicate to the intermediate nodes that the sender does not yet have a valid estimate of the round trip time rtt. The sender also initializes the fb field to such that if bandwidth is available, this initialization allows the sender to reach the desired rate after one ci. When the first ACK returns, the sender calculates rtt and ci according to the method described next, sets $r_p = 1/rtt$, and sends out the second data packet. Thereafter, RBCC sends out a data packet only when the transmission window allows and a permit is available.

The RBCC sender maintains an estimate of the smoothed round trip time $srtt$ and calculate the control interval ci as

$$ci = max(srtt, 5/r_p). \tag{3}$$

When r_p is large, i.e., $r_p > 5/srtt$ and $ci = srtt$, the sending rate will reach the path capacity after one $srtt$. Otherwise, this period equals $5/r_p$. The value of the control interval thus ensures that, on average, there are at least 5 data packets being transmitted in this period. If the period is too long, the adjustment of the sending rate is sluggish to respond to the load change along the path. If the period is too short, the estimation of the feedback over short intervals at the nodes along the path leads to erroneous estimates, and sometimes there may be no feedback received in one control interval. The choice of 5 is the tradeoff between these two considerations.

RBCC adjusts r_p according to the fb in the ACK packet whenever a new ACK arrives, and

$$r_p = r_p + fb. \tag{4}$$

The nodes along the path control the feedback such that the sending rate is allowed to reach the path capacity after one control interval ci.

Retransmission timer RTO will expire when there is a packet loss. Notice that, in MANETs, queue overflow rarely

happens for TCP flows. And packet losses mainly result from the failed transmission attempts at the MAC layer due to contention, collision, wireless channel error, or mobility-caused route failures. Subsequently, the link breakage will be reported to the routing protocol, which may further drop subsequent packets. Notice in this case, the original route is broken, thus the timeout signals not only the packet loss, but also the route breakage. To avoid long periods of pausing and hence waste of channel capacity, it is wise for RBCC to send out a probe message or just retransmit the lost packet in periodic intervals to detect whether a new route is established.

Therefore, the response of RBCC to timeout is the following. For the first timeout, the RBCC sender retransmits the corresponding packet, double the retransmission timer, and reset r_p to $1/RTO$. Note retransmitted packets have higher priority than normal packets. In other words, the retransmitted packets will be transmitted when the next permit arrives, no matter whether there are any other packets in the window. For the subsequent back-to-back timeouts before a new acknowledgements arrives, RBCC does not double its retransmission timer again, and nor does it reset r_p. It also records the time when the first timeout in the window takes place to differentially treat the feedback information carried by the ACKs that arrive after the timeout and route repair. The feedback in those ACKs that acknowledge the packets that are sent prior to the timeout is simply ignored, since it is very likely the feedback was calculated before the route failure and hence becomes outdated. By contrast, the feedback in those ACKs that acknowledge the packets that are sent later than the timeout are used to adjust the permit arriving rate as normal.

C. Feedback Calculation

To achieve both the efficiency and fairness, RBCC relies on the calculation of the explicit feedback in each average control interval denoted by $\overline{ci}$. Specifically, in each $\overline{ci}$, each node first determines the aggregate feedback that is defined as the change in the total channel throughput at the MAC layer; then, the node will fairly allocate the change in the throughput to each flow traversing the node.

Each node maintains a per-node estimation-control timer that is set to the most recent estimate of $\overline{ci}$. It is calculated by using the information in the congestion header. Upon timeout the node updates its estimates and its control decisions.

In each $\overline{ci}$, a node monitors all received and transmitted packets and accordingly update total input and output traffic S and the metrics S_1 and S_2. And,

$$\begin{aligned} S &= \sum_j l_j, \\ S_1 &= \sum_j \frac{1}{r_{pj}} = \sum_k \frac{\overline{ci} \times r_{pk}}{r_{pk}} = \overline{ci} \times K, \\ S_2 &= \sum_j \frac{ci_j}{r_{pj}} = \overline{ci} \sum_k ci_k, \\ K &= \frac{S_1}{\overline{ci}}, \quad \overline{ci_{new}} = \frac{S_2}{S_1}, \end{aligned} \tag{5}$$

where j is the index for each packet, l_j is the length in bytes of the jth packet, k is the index for each flow, K is estimated total number of flows, $\overline{ci} \times r_{pk}$ is the estimated total number of transmitted packets of flow k in $\overline{ci}$, $\overline{ci_{new}}$ is the new estimated value of $\overline{ci}$. Notice that in all the summations in Equation 5, the node will update S, S_1 and S_2 whenever it receives or transmits a packet because both receiving and transmission consumes the shared channel resource. For instance, at node 3 in Fig. 3, $S = r_1 + r_2 + 2 \times r_3$ where $r_i (1 \le i \le 3)$ is the traffic of flow i, and $K = 4$ which will be used in our fair allocation algorithm.

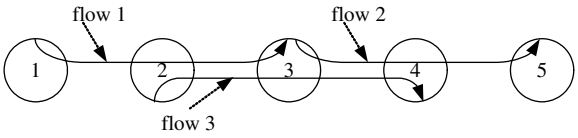

Fig. 3. Chain topology with 5 nodes

To achieve efficiency, i.e., maximizing the channel utilization while avoiding severe MAC layer contention, RBCC calculates the aggregate feedback ΔS in each $\overline{ci}$ as follows:

$$\Delta S = \frac{th_b - r_b}{r_b} \times S, \tag{6}$$

After calculating ΔS, the change in the total traffic or the aggregate feedback, RBCC needs to apportion it to individual flows in order to achieve fairness. We denote per packet feedback in bytes/s for packet j as pf_j, and

$$\Delta S = \sum_j pf_j \times \overline{ci}, \qquad fb_j = \frac{pf_j}{l_j} \tag{7}$$

RBCC relies on an *Additive-Increase Multiplicative-Decrease (AIMD)* policy to converge to fairness: If $\Delta S > 0$, all flows increase the same amount of throughput. And if $\Delta S < 0$, each flow decreases the throughput proportionally to its current throughput.

Thus, if $\Delta S >= 0$, the increasing amount of traffic rate for each flow C_p, and per packet feedback pf_j will be

$$C_p = \frac{\Delta S}{\overline{ci} \times K}, \qquad pf_j = \frac{C_p}{r_{pj} \times \overline{ci}} \tag{8}$$

Note that for those flows that either originate or terminate at the node, the node counts each as one flow, whereas for those flows only pass the node, the node counts each as two flows, i.e., one in and one out. This is because the flow passing the node contributes to both the input traffic and the output traffic.

If $\Delta S < 0$, the per packet feedback is proportional to the throughput, i.e., $r_{pj} \times l_j$, and inversely proportional to the expected number of packets seen by the node in $\overline{ci}$, i.e., $r_{pj} \times \overline{ci}$. Thus according to Equation (7),

$$pf_j = C_n \frac{r_{pj} \times l_j}{r_{pj} \times \overline{ci}} = \frac{C_n \times l_j}{\overline{ci}}, \qquad C_n = \frac{\Delta S}{S} \tag{9}$$

IV. Performance Evaluation

In this section, we demonstrate through simulations that RBCC outperforms TCP in the MANET. In contrast to TCP, the new protocol dampens the oscillations of channel utilization, quickly converges to high utilization, short round trip time, small queue size, and fair bandwidth allocation.

A. Simulation Settings

We use network simulator ns-2 to conduct the simulations. In ns-2, the physical-layer propagation model is two-ray ground model. We adopt the default values in ns-2 for the transmission power and other physical parameters, which indicate the transmission range is about 250m and the interference range is about 550m. The channel bandwidth is 2Mbps. The payload size of each DATA packet is 1000 bytes. The default buffer size at each node is 50.

In the simulation, a chain topology is used because it is simple and allows us to clearly demonstrate the problems and the advantages of RBCC over TCP. Under this topology, we consider two routing schemes. The first one uses the pre-computed shortest path, therefore has no routing overhead, which enables us to focus on the MAC and the transport layer and observe their interaction. The second one is AODV, which enables us to show how on-demand routing schemes affect the performance of RBCC and TCP.

B. Simulation Results

The chain topology consists of 9 nodes as shown in Fig. 1. Nodes are separated by 200m. The pre-computed shortest path routing is used. In this simulation, there are 6 identical flows. The source is node 1 and the destination is node 9. The aggregate throughput, end-to-end delay, and packet dropping rate is presented in Table III.

TABLE III

PERFORMANCE OF RBCC AND TCP IN CHAIN TOPOLOGY

	Pre-computed Path		AODV	
	TCP	RBCC	TCP	RBCC
Throughput(Kb/s)	204.70	230.84	128.88	202.68
End-to-end delay(s)	0.6265	0.06914	0.6330	0.1319
Dropping(pkt/s)	1.2759	0.0379	0.6138	0.082

In Fig. 4 (a), the channel busyness ratio is presented. Each point in the curves is an average value during each second. It can be observed that RBCC quickly converges to high link utilization and remain stable, while TCP always oscillates in a large range. We also observe that different nodes see different channel busyness ratio. Since node 5 is in the middle of the chain and thus encounters the most serious collisions, it's channel busyness ratio is the largest. On the other hand, compared to node 1, node 9, as a destination, does not transmits any DATA packets, so it observes the smallest channel busyness ratio.

Fig. 4 (b) demonstrates that RBCC has a much smaller round trip time, rtt, than TCP. For RBCC, the average value of rtt is $0.1216s$, as opposed to $1.12s$ for TCP. Given that the measured throughput for each RBCC flow is 38.5 Kbits/s, the transmission time for one packet is $0.22s$, which is larger than the average rtt. In other words, the window size for RBCC is less than 1. This is consistent with our previous analysis that the optimum window size may be less than one. For TCP, normally its window size is larger than one packet per rtt.

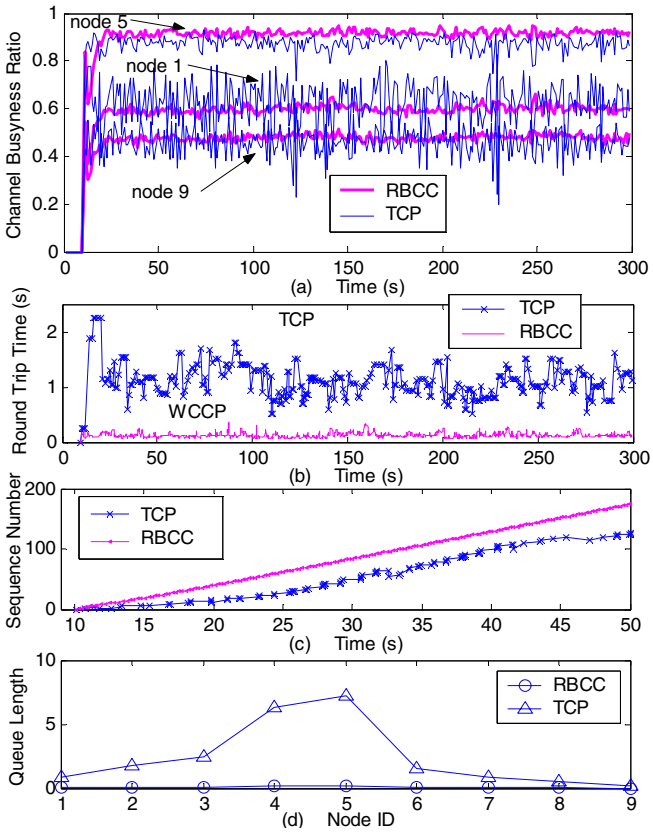

Fig. 4. Chain topology with 6 identical flows from node 1 to node 9 using the pre-computed shortest path

Since TCP overestimates the window, it is no wonder that TCP incurs severe congestion and MAC collisions in MANETs, or the packet dropping rate due to MAC collisions is about 33 times as high as that of RBCC traffic, as shown in Table III.

In Fig. 4 (c), we see that as the sequence number of received packets is smoothly increasing, each RBCC flow has a stable throughput, and almost no packet needs to be retransmitted. Here the instantaneous throughput of a flow can be calculated by taking derivative of the sequence number with respect to time, i.e., the slope of the curve in Fig. 4 (c). This further verifies that RBCC almost does not drop packets due to MAC collision or buffer overflow. On the contrary, TCP frequently retransmit lost packets. For clear demonstration, we only present the result from the start of the transmission, namely $10s$, to $50s$, although we observe the similar phenomenon for the rest of the simulation. As a result, the throughput of RBCC is 12.8% higher than that of TCP, as shown in Table III.

Fig. 4 (d) shows that RBCC maintains a much smaller queue size at all the nodes than TCP. In fact, the average queue size of RBCC is always smaller than 1. This translates into a short end-to-end delay for RBCC, which is only 1/9 of that of TCP, as shown in Table III. In addition, as pointed out earlier, a large queue size keeps node busy with contending the channel, which increases contention and causes packets to be dropped. Thus, a small queue size is desirable. This also explains why

TCP has a much larger packet dropping rate (in packets/s) than RBCC.

We also simulate the same topology with the AODV routing algorithm. As seen in Table III, the aggregate TCP throughput drops 37% compared with the case where the pre-computed shortest path is used. However, the aggregate RBCC throughput only drops 12%. As a result, in this case, the aggregate throughput of RBCC is 57% higher than that of TCP.

V. Conclusions

Congestion control is critical to reliable transport service in MANETs. Traditional TCP suffers severe performance degradation and unfairness. Realizing the core cause is due to the poor interaction between traditional TCP and the 802.11 MAC, we propose a systematic solution named Wireless Congestion Control Protocol (RBCC) to address this problem. The major contributions of this work is three-fold. First, we use simulation studies to show that window-based congestion control mechanism, say that of TCP, results in poor and unstable performance due to unique medium contention in MANET and hence argue that rate-based congestion control may be more appropriate for MANET. Second, we propose to use channel busyness ratio, which is a good sign of network congestion and available bandwidth at the MAC layer, to calculate explicit and precise feedback at the transport layer. Third, we propose an end-to-end congestion control protocol, which uses channel busyness ratio to adjust the sender's rate so that the channel capacity can be fully utilized and fairness is improved. Simulation results show that our scheme significantly outperforms traditional TCP in terms of channel utilization, end-to-end delay, and fairness.

Acknowledgement

This work was supported in part by the U.S. Office of Naval Research under Young Investigator Award N000140210464 and under grant N000140210554.

References

[1] K. Chen, Y. Xue, and K. Nahrstedt. On setting TCP's congestion window limit in mobile ad hoc networks. In *Proc. IEEE ICC 2003*, May, 2003.

[2] X. Chen, H. Zhai, J. Wang and Y. Fang. TCP performance over mobile ad hoc networks. *Canadian Journal of Electrical and Computer Engineering*, Vol. 29, No. 1/2, p129-134, January/April 2004.

[3] Z. Fu, X. Meng, and S. Lu. How Bad TCP can Perform in Mobile Ad-Hoc Networks. In *IEEE Symposium on Computers and Communications*, July 2002

[4] Z. Fu, P.Zerfos, H. Luo, S. Lu, L. Zhang, M. Gerla. The Impact of Multihop Wireless Channel on TCP Throughput and Loss. In *proc. IEEE INFOCOM*, March 2003.

[5] M. Gerla, R. Bagrodia, L. Zhang, K. Tang, and L. Wang. TCP over Wireless Multihop Protocols: Simulation and Experiments. In *Proc. IEEE ICC*, June 1999.

[6] M. Gerla, K. Tang, and R. Bagrodia. TCP Performance in Wireless Multihop Networks. In *Proc. IEEE WMCSA*, Feb. 1999.

[7] S. Xu and T. Safadawi. Does the IEEE 802.11 MAC Protocol Work Well in Multihop Wireless Ad Hoc Networks? *IEEE Communications Magazine*, pp. 130-137, June 2001.

[8] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash. A feedback-based scheme for improving TCP performance in ad hoc wireless networks. *IEEE Personal communications*, 8 (1):34-39, Feb. 2001.

[9] G. Holland and N. H. Vaidya. Analysis of TCP performance over mobile ad hoc networks. In *Proc. ACM MOBICOM*, Aug. 1999.

[10] J. P. Monks, P. Sinha and V. Bharghavan. Limitations of TCP-ELFN for ad hoc networks. In *Proc. MOMUC*, 2000.

[11] J. Liu and S. Singh. ATCP: TCP for mobile ad hoc networks. *IEEE JSAC*, Vol.19 No.7, July 2001.

[12] T. D. Dyer and R. V. Boppana. A comparison of TCP performance over three routing protocols for mobile ad hoc networks. In *Proc. ACM Mobihoc*, Oct. 2001.

[13] F. Wang and Y. Zhang. Improving TCP performance over mobile ad-hoc networks with out-of-order detection and response. In *Proc. ACM MobiHoc*, June 2002.

[14] K. Xu, M. Gerla, L. Qi, and Y. Shu. Enhancing TCP fairness in ad hoc wireless networks using neighborhood RED. In *Proc. ACM MobiCom*, Sep. 2003.

[15] Y. Afek, Y. Mansour. Z. Ostfeld, Phantom: a simple and effective flow control scheme. In *Proc. ACM SIGCOMM*, 1996.

[16] D. Katabi, M. Handley, and C. Rohrs. Congestion control for high bandwidth-delay product networks. In *Proc. ACM SIGCOMM*, 2002.

[17] *IEEE standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications*, ISO/IEC 8802-11: 1999(E), Aug. 1999

[18] The network simulator ns-2. http://www.isi.edu/nsnam/ns.

[19] J. Li, C. Blake, D. S. J. De Couto, H. I. Lee and R. Morris. Capacity of ad hoc wireless network. In *Proc. ACM MobiCom*, July 2001.

[20] C.E. Perkins, E.M. Royer, and S.R. Das. Ad Hoc On Demand Distance Vector (AODV) Routing. IETF RFC 3561

[21] C.E. Perkins, E.M. Royer, S.R. Das, and M.K. Marina. Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks. *IEEE Personal Communications*, pp. 16-28, Feb. 2001.

[22] K. Chen, and K. Nahrstedt. Limitations of Equation-based Congestion Control in Mobile Ad hoc Networks. In *Proc. International Workshop on Wireless Ad Hoc Networking (WWAN 2004) in conjuction with ICDCS-2004*, March, 2004

[23] S. Floyd, M. Handley, J. Padhye, and J. Widmer. Equationbased congestion control for unicast applications. In *Proc. ACM SIGCOMM*, Aug. 2000.

[24] H. Zhai, X. Chen, and Y, Fang. How Well Can the IEEE 802.11 Wireless LAN Support Quality of Service? Accepted for publication in *IEEE Transaction on Wireless Communications*, 2004.

[25] H. Zhai, X. Chen, and Y. Fang. A Call Admission and Rate Control Scheme for Multimedia Support over IEEE 802.11 Wireless LANs. In *Proc. First International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks (QShine'04)*, Oct. 2004.