Enhanced Rate-based Transmission control for TCP in ad hoc Networks

Mohamed Yahia

Department of Telecommunications and Media Informatics,
Budapest University of Technology and Economics,
H-1117, Budapest, Magyar Tudosok krt. 2

Summary
This paper studies TCP performance over multi-hop wireless ad hoc networks that use the IEEE 802.11 protocol as the access method. The aim is to improve the TCP fairness while keeping the algorithm as simple as possible, since in previous works the algorithm designs were more complicated. We propose a simple approach to improve fairness based on scheduling (pacing) new packets according to the transmission interval formed from scaled round-trip time (RTT) and congestion window. Our simulation using static and mobile networks shows that, given specific scale parameter *x*, TCP achieves high fairness and throughput via improved spatial channel reuse, if it operates in certain range of the transmission interval.

Key words:
Ad hoc networks, Wireless networks, Routing protocol, TCP protocol, Hidden terminal.

1. Introduction

In packet-radio networks a Medium Access Control (MAC) protocol is essential so that stations can share a common broadcast channel. Carrier Sense Multiple Access (CSMA) protocols have been used in a number of packet-radio networks. The goal of these protocols is to prevent multiple stations from transmitting simultaneously within their range, by listening on the channel before transmitting. CSMA can not solve hidden-terminal problems which cause degradation in the performance, however, because it cannot prevent collisions.

Let us first identify two critical issues leading to the TCP performance degradation: (1) unreliable broadcast, since broadcast frames are transmitted without the request-to-send and clear-to-send (RTS/CTS) dialog and Data/ACK handshake, so they are vulnerable to the hidden terminal problem; and (2) false link failure which occurs when a node cannot successfully transmit data temporarily due to medium contention. A hidden terminal is a node in the receiver's neighborhood which cannot detect the sender and may disrupt the current packet transmission [1]. Recently, several techniques have been proposed to improve TCP performance in ad hoc networks and solve hidden terminal problems. Most of these techniques address mobility, link breakages and routing algorithm failures [15], [12], [13], [2], [8], [14].

In this paper we introduce a method to improve TCP performance over ad hoc networks which can solve the hidden terminal problem if it exist. M-TCP (Modified TCP) is a novel and simple congestion control algorithm with TCP over multi-hop IEEE 802:11 networks. It implements rate-based scheduling of transmission within the TCP congestion window. Rather than adaptively estimating the 4-hop propagation delay and the coefficient of variation of recently measured round-trip times as in [3], in our approach a TCP sender sets its transmission interval for the current congestion window and round-trip time using a scaling parameter. The useful range of the scaling parameter can be identified in a straightforward manner. A comprehensive study using the NS2[4] shows that M-TCP achieves a fairness greater than 99%, provides high throughput in almost all scenarios, and is highly responsive to changing network traffic. Our approach provides each node with a fair share of the available bandwidth, even when flows are not within each other's transmission range but are within each other's interference range. M-TCP also requires no modifications on the routing layer or the link layer.

2 Backgrounds and Related Work

TCP is a connection-oriented transport layer protocol that provides reliable in order delivery of data to a TCP receiver. Since the characteristics of wire-line and wireless networks differ, a TCP algorithm which is designed to perform well in wire-line networks suffers from degradation in wireless networks.

2.1 Why TCP inefficient in wireless networks

There are several reasons for such failure for TCP throughput in ad hoc wireless networks. In this section we discus some of these reasons in details.

2.1.1 High Bit Error Rate

One of the problems with TCP in ad-hoc networks is the effect of a high bit error rate, which may corrupt packets and result in lost data segments or acknowledgments (ACKs). If an acknowledgment is not received by the

Manuscript received January 5, 2008
Manuscript revised January 20, 2008

sender within the retransmission time out (RTO) window, the sender does the following:

- Retransmits the segment.
- Exponentially backs off its retransmission timer.
- Reduces its congestion control window
- Closes its congestion window to one segment.

In the case of repeated errors the congestion window at the sender will remain small, resulting in low throughput.

When hidden terminals exist in the network and the destination node is located in their interference range, a sender node (B) that receives no response after seven retransmissions of the RTS will drop its head-of-line packet according to the IEEE 802.11 MAC layer. Waiting for a period of time longer than the RTO results in the TCP sender timing out, retransmitting a packet and invoking congestion control.

2.1.2 Serial Timeouts

The serial timeout is a condition wherein multiple consecutive retransmissions of the same segment are transmitted to the receiver while it is disconnected from the sender. This happens when their transmission timer at the sender is doubled with each unsuccessful retransmission attempt in order to reduce the transmission rate. Thus, when the mobile is reconnected, TCP will take a long time to recover from such a reduction and data will not be transmitted for a period of time.

2.1.3 Route Recomputations

When the route is no longer available, the network layer at the sender attempts to find a new route to the destination. If discovering the new route takes longer than RTO at the sender, the sender timeouts, retransmits a packet and invokes congestion control. Thus, when a new route is discovered, the throughput will continue to be small for some time due to the behavior of congestion window when uses slow start and congestion avoidance. This makes TCP inefficient if it is done frequently (due to high mobility).

2.2 TCP Westwood

TCP Westwood (TCPW) [5] is essentially designed to improve the performance of TCP Reno in both wired and wireless networks. TCPW relies on end-to-end bandwidth estimation to determine the cause of packet loss (congestion or wireless channel effect), which is a major problem in TCP Reno. The idea is to continuously evaluate the quality of the connection at the TCP source by monitoring the rate of returning ACKs, and set the congestion window and slow start threshold (ssthresh) accordingly. Although TCPW performs well in ad hoc wireless networks, it still suffers from unfairness when hidden terminals are present.

2.3 Related Work

TCP Adaptive Pacing (TCP-AP) is another version of the TCP protocol which tries to solve the hidden terminal problem in ad hoc wireless networks. Sherif et al. [3] introduced TCP-AP, which implements rate-based scheduling within TCP's congestion window in order to avoid burst packet transmissions and quantify incipient congestion. It measures the fluctuation (coefficient of variation) in a sample of round trip times and adaptively calculates the appropriate pacing of transmission. TCP-AP schedules new packets according to this computed rate, and allows the node to delay transmission of a data packet until the previously sent packet has been forwarded 4 times. Using simulations the authors showed that TCP-AP could substantially improve goodput with respect to TCP NewReno.

Singh et al.[6] developed a MAC protocol that employs adaptive interference cancellation based on cross-layer design considerations to increase network throughput and fairness. They used simulations with different tools and conclude that in terms of providing throughput gains and energy savings, it is better to exploiting multiuser diversity by interference cancellation than to use multiple antennas with an IEEE 802.11 MAC.

Chane et al. [7] analyzed Floor Acquisition Multiple Access (FAMA) protocols for single-channel packet-radio networks with hidden terminals. These protocols permit a station to acquire control of the channel dynamically before transmitting data packets. Their verification and throughput analysis, also supported by simulations, demonstrated that carrier sensing significantly improves performance in single channel networks in the presence of hidden terminals.

As opposed to the approaches mentioned above, our enhanced TCP requires neither modifications to the routing or link layers, nor cross-layer information from intermediate nodes along the path. Our M-TCP approach also fundamentally differs from TCP-AP [3]. Although M-TCP does depend on the delay of transmission packets, the computation of transmission intervals is based on the measured congestion window and round-trip times rather than using 4-hop transmission delay and the RTT coefficient of variation.

3 Rate-Based Transmission of Packets

3.1 Motivation

In IEEE 802.11, control handshake Request-To-Send/Clear-To-Send (RTS/CTS) messages precede each packet transmission. Due to the spatial reuse constraint of the wireless channel, neighboring nodes of both sender and receiver defer their transmission until the subsequent DATA-ACK transmission is completed. Thus packet bursts caused by TCP's window-based congestion control result in increased contention on the wireless channel. This contention on the link layer may lead to packet drops due to the hidden terminal problem [8]. Our TCP algorithm focuses on improving fairness in the presence of a hidden terminal. Since the sender can obtain rate information from the feedback packets of the receiver, the packet transmission in these networks can in principle be adapted to the rate. Our goal is to incorporate a rate-based transmission algorithm into a window-based TCP congestion control.

To this end, for each received ACK, TCP computes the transmission interval according to the current congestion window and the measured round-trip time *RTT* divided by a scaling parameter *x*. It then schedules the transmission of new packets based on this interval. This creates a rate-based congestion control which is sufficiently simple for efficient implementation.

In current TCP variants such as Reno and NewReno, congestion is managed solely upon the observation of packet losses. Although Vegas uses both RTT and packet losses to identify congestion, it still suffers from the negative effect of packet transmission bursts in wireless multihop networks. These characteristics of IEEE 802.11 make it obvious that TCP suffers from poor performance. Thus, the aim of this work is to develop a congestion control that identifies high contention on the network path of the TCP connection and throttles the transmission rate before losses occur. In order to improve the fairness of TCP on the network path, we propose a transmission interval adjustment based on the state of the congestion window *CW*, the recently measured round-trip time *RTT*, and a scale parameter *x*. This transmission interval is given by the formula:

$$T_{\text{int}\,erval} = RTT * 1/x * CW \qquad (1)$$

3.2 The Effect of the Scale Parameter *x* on M-TCP Performance

According to equation 1, a higher values of parameter *x* result in a smaller $T_{\text{int}\,erval}$ which in turn scheduling the transmission of the packets according to this interval, improve the spatial reuse, and improve the TCP throughput. Although improving TCP throughput is required, a small fraction of fairness is also sacrificed. In this paper we consider which values of *x* result in both a high level of fairness and acceptable throughput. After run several simulations with different values of scaled parameter *x*, we deduced that, higher throughput and excellent fairness may produced when *x* becomes very close to value 50. Some values < 50 have also high fairness but lower throughput. The amount of throughput that may be sacrificed when hidden terminal exist does not exceed 15% compared with TCP-AP. (see section 5 Figure 14)

3.3 The Hidden Terminal Problem

A hidden terminal is a potential sending node in the receiver's interference range which *cannot* detect the sender, and thus may disrupt transmission of the current packet. Although CSMA is designed to solve the collision problem in an IEEE 802.11 wireless network by RTS/CTS handshake messages, a hidden terminal degrades its performance substantially. Carrier sensing simply cannot prevent these collisions.

Let us assume that we have chain network topology as in Figure 1, where node 1 wishes to transmit to node 2 and node 4 wishes to transmit to node 5. In this case, node 4 is a hidden terminal for node 2 because it can neither receive the RTS/CTS handshake between nodes 1 and 2 nor sense the transmission from node 1 to node 2. Since node 1 is out of the sensing range of node 4, it can transmit to node 2 while node 4 transmits to node 5. Thus, a collision occurs at receiving node 2, since it is in the interference range of node 4, which leads to contention loss.

We can also examine the hidden terminal problem in a cross topology (Figure 2). Assume that node 2 transmits to node 9, and node 6 transmits to node 4. In this case node 4 becomes a hidden terminal for the transmission from node 2 to node 9.

In our approach, the hidden terminal problem can be avoided by adjusting the transmission interval of packets ($T_{\text{int}\,erval}$).

3.4 Unfairness due to Hidden Terminals

The Distributed Coordination Function (DCF) in IEEE 802.11 defines two methods in accessing the medium; the two-way and four-way handshake. The two-way handshake occurs when sender transmits the data to the receiver and the receiver responds with an ACK if the data received successfully. In the four-way handshake, the sender first transmits out the RTS and responding to this; the receiver sends out CTS if it found the idle medium, after that the sender sends the data and receiver responds with an ACK.

Every node in IEEE 802.11 allowed to maintain a contention window (*w*) and a back-off timer according to the well-known Binary Exponential Back-off (BEB) algorithm used. Whenever node wants to transmit, it defers by back-off timer which generated according to this formula:

$$Backoff - Time = Random() * SlotTime$$

Where *Random* value is uniformly distributed over the range [0, *w*] and the *SlotTime* is specified by the physical layer. The new back-off timer value is generated whenever becomes zero. At the first transmission of the packet the *w* will be set to $w_{\min}$ and doubles whenever retransmission is initiated. When retry limit is reached or in case of successful transmission occurs the *w* will be reset to $w_{\min}$. Now let us see what happen when node (say 4 in Figure. 1) wants to transmit. In this case, all the nodes in its range will *freeze* their back-off timers until the transmission completes and the medium becomes an idle. Then, all the nodes defer for DCF Inter-Frame Space (DIFS) period. While the node 4 generates new random value and backs off before it initiates another transmission, the other nodes resume to count down from their frozen back-off timers. Thus, node 4 may transmit several packets before another node (e.g. node 2 when it sends CTS to node 1) back-off timer is reduced to zero. This is unfair for node 1 and may lead to starvation if it repeated several times [9].

4 Simulation Setup and Results

The results reported in this paper are obtained using the NS2 network simulator.

4.1 Setup Environment

The simulation parameters are given in Table 2. We simulated two static (fixed) networks of *i* and *k* nodes in the linear chain and cross configurations respectively (Figures 1 and 2). For each network we consider the performance in the absence and presence of a hidden terminal. We also simulate our approach with mobile network and generate 30 nodes randomly moving in an area of 1750x1500m. These nodes move to random positions in random speed uniformly distributed with average of 5m/s. The specific simulation scenarios are shown in Table 1.

In order to test our approach, we established two TCP connections with 1 kB packets continuously transferred by File Transfer Protocol (FTP). In all network scenarios, each node is separated by 250 meters from its adjacent nodes. Our ad hoc routing protocol is On-Demand Distance Vector Routing (AODV) [10]. Each simulation ran for 200 s, and the results have been plotted in Figures. All simulation parameters which are not explicitly stated in this paper are set to the *NS2* defaults.

Table 1: Simulation scenarios

No	*Scenario*
1	Node 1 send to 2 and 5 to 6, chain, static
2	Node 1 send to 2 and 4 to 5, chain, static
3	Node 1 send to 5 and 6 to 9, cross, static
4	Node 2 send to 9 and 6 to 4, cross, static
5	Random of movement and speed, mobile

Table 2: Simulation setup parameters

Parameter	*Type*
Transmission range	250m
Interference, sensing range	550m
Propagation model	Two-Ray ground
Antenna model	Omni-directional
Channel bandwidth	2Mbit/s

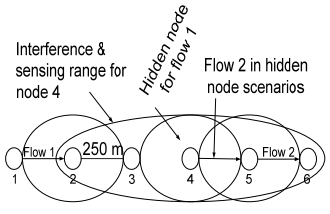

Fig. 1 Chain Topology

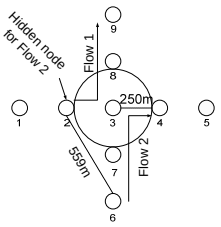

Fig. 2 Cross Topology

4.2 Results

4.2.1 M-TCP Fairness

We begin by computing Jain's fairness index [11] for each parameter set:

$$F(bw_i, i=1,..,n) = \left(\sum_{i=1}^{n} bw_i\right)^2 / n \cdot \sum_{i=1}^{n} bw_i^2 \qquad (2)$$

Where bw_i are the bandwidth shares of the TCP flows. Figure 3 shows the impact of x on the fairness of the network.

Chain Network: In Scenario 1 there is no contention due to a hidden terminal, so both flows take up equal shares of the available bandwidth. Thus, most TCP algorithms have a high fairness in this case. As shown in Figure 3. M-TCP achieves a fairness between 99% and 100% for all $x \leq 50$. The situation is different in Scenario 2, where TCP suffers from unfairness due to the interference caused by hidden terminals and the performance of the network degrades. M-TCP, however, continues to achieve a fairness index of more than 99% in this case for all values of x.

Cross Network: In the cross network, flows share the forwarding node at the intersection. The fairness thus fluctuates around specific points. As shown in Figure 3, the highest fairness levels were recorded for $x = 35$ in both Scenario 4 and Scenario 3 (98% and 97% respectively). Other values of x also achieve a fairness index of 95% or better.

Random Network: Even in mobility (Scenario 5), M-TCP proved a high response and achieved 99% of fairness for all x values as shown in Figure 3.

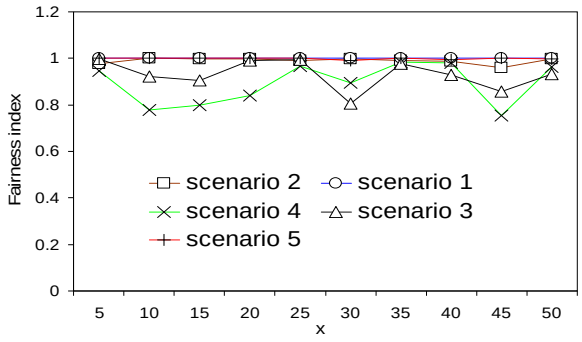

Fig. 3 M-TCP Fairness

4.2.2 M-TCP Throughput

Although network fairness is the main interest of this paper, we also wish to investigate how well M-TCP achieves spatial channel reuse (since improving spatial reuse increases TCP throughput). In Figures 4, 5 and 6 we show both TCP flows in-order to observe the fairness. The throughput is shown as a function of the scaling parameter x for the chain, cross and mobile networks respectively. We observe that M-TCP flows share the available bandwidth equally in networks containing a hidden terminal, rather than allowing one flow to dominate as in other TCP variants (TCPW, TCP-AP, Standard TCP; see Figures 9 to 12).

Chain Network: In scenario 1, the flows are identical and the simulation shows that throughput increases as long as x increased (see Figure 4) where it reaches the other TCP variants as will shown in Section 5. In contrast, the flows in Scenario 2 are ``twisted''. At $x = 50$, however, M-TCP throughput exceeds the standard TCP and TCPW by 1.5% or more (see Figure 14 in Section 5).

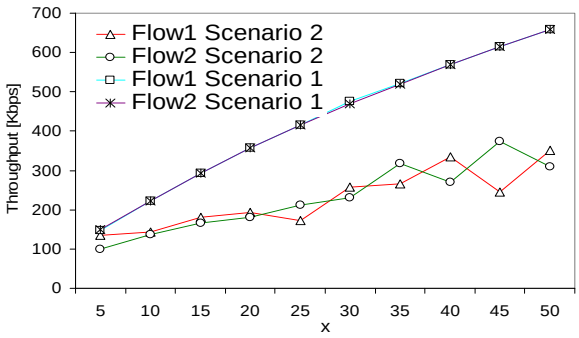

Fig. 4 Throughput in chain network

Cross Network:

In order to test the influence of a hidden terminal in the cross network we investigate Scenario 4 where node 4 (receiver side of flow 2) is under the effect of interferences of node 2. In Figure 5, M-TCP improves the throughput flow 2 and achieves high fairness only for certain values of the scaling parameter ($x = 25$, 35, 40, and 50); note that these values are also evident in Figure 3. In Scenario 3, both flows increase their throughput gradually with increasing x but in a "twisted" manner. When x is 50, the M-TCP throughput improves to 161 Kbps. This is 5% more than that the value achieved by TCPW and standard TCP (152 Kbps). The maximum fairness achieved by M-TCP is also 99%, better than the other TCP variants (see Table 3).

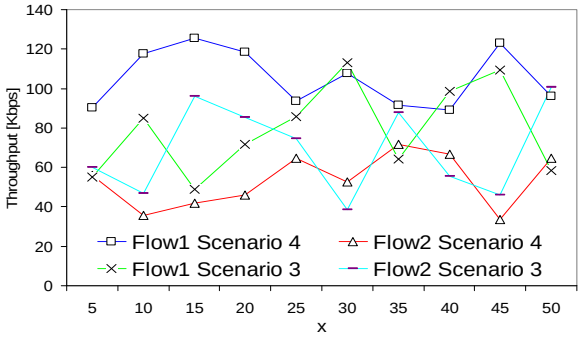

Fig. 5 Throughput in cross network

Random Network:
In mobile networks, TCP throughput usually lies in problems mentioned in section 2. Although the flow 1 becomes under the effect of hidden terminal at second ≈130 (see Figure. 13), M-TCP throughput approximately increases with increasing *x* in linear manner as shown in Figure 6.

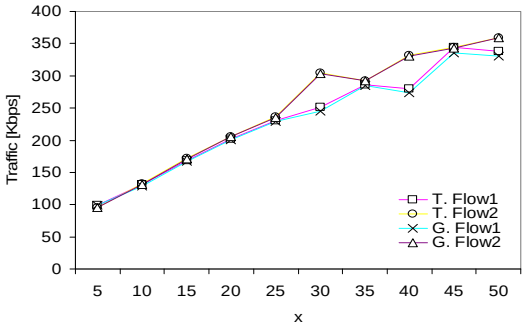

Fig. 6 Throughput(T) and Goodput(G) in scenario 5

4.2.3 M-TCP Goodput and Loss

In a hardwired Internet, packet loss is mainly due to buffer overflow at the bottleneck router. In a multihop wireless network, on the other hand, packet drops may be caused by both buffer overflow and contention due to hidden terminals. In Figures 7 and 8, we present the goodput and the loss of all scenarios for those values of *x* where the network achieved the highest fairness index. We can observe an approximately linear increase of goodput with increasing *x* in scenarios 1 and 2 (Figure 7). The loss in both of these scenarios (Figure 8) is also quite small; even in Scenario 2, its maximum value is 3.35% at $x = 45$. The goodput levels in scenarios 3 and 4 are very close to each other, and do not appear to depend strongly on *x*. Choosing the right value of *x* can still minimize the loss, however. In scenario 4, for example, choosing $x = 45$ results in a low loss rate of 4.5%. In scenario 5, the aggregate goodput shown in Figure 7 is similar to that in scenario 2. The loss is small too but increases with increasing *x*.

For further improvements we have also undertaken a preliminary investigation of dropped packets. We find that in all scenarios most of the dropped packets are due to MAC layer collisions. As an example, consider the cross network without hidden terminals (scenario 3) at $x = 5$, where the highest rate of dropped packets occurred (Figure 7). In this case we find that 12% of lost packets were dropped due to the callback in routing protocol, 77% were due to collisions in the MAC protocol, 6% were caused by route failure, 4% were due to retransmission time out, and 1% were dropped because of buffer size. Even this loss, however, can be reduced to lower rates by choosing larger *x* but on the expense of fairness.

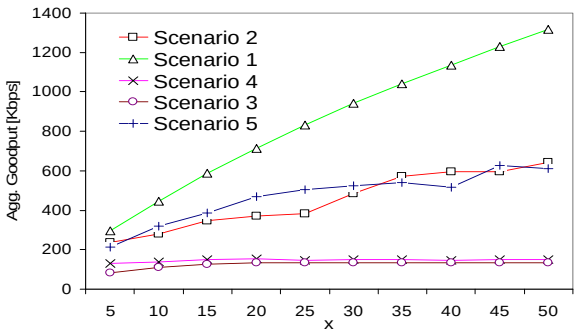

Fig. 7 Goodput for all scenarios

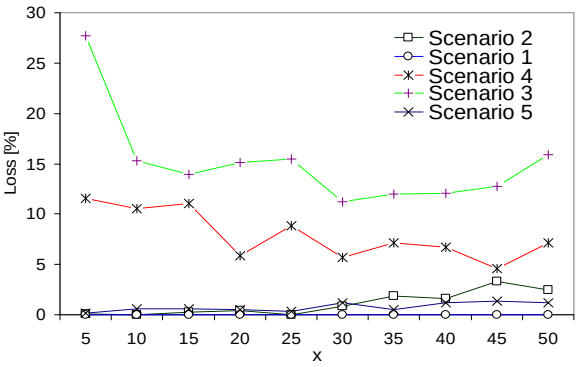

Fig. 8 Rate of loss

5 Comparison with other TCP versions

We now compare the fairness obtained by M-TCP, TCP Westwood, TCP-AP, and Standard TCP for all four scenarios (Table 3). Note that M-TCP is always the fairest algorithm, sometimes by a substantial amount. In Figures 9 through 12, we compare the behavior of the congestion window over time under M-TCP, TCP Westwood, and TCP-AP and omit that related to Standard TCP because it is identical to TCP Westwood. Since we are mainly concerned with hidden terminal problems, only those results obtained in Scenarios 2, 4 and 5 are shown.

Table 3 M-TCP and other TCPs fairness in [%]

	Sce 1	*Sce 2*	*Sce 3*	*Sce 4*	*Sce 5*
TCPW	100	50	94	86	87
TCP-AP	100	56	98	69	91
M-TCP	100	99	99	97	99
St.TCP	100	50	61	89	87

In Scenario 2 (Figure 9), it is clear that under TCP Westwood flow 1 never had the opportunity to transmit at the maximum negotiated rate, because its congestion window was always significantly smaller than the advertised window size. The other flow thus dominated the channel bandwidth in this case. Under TCP-AP (Figure 11), flow 1 made some attempts at transmission but never achieved a significant bandwidth share.

In Scenario 4 (Figure 10), it is obvious that both TCP Westwood flows had better opportunity to transmit and this is also clear from the high fairness shown in Table 3. In TCP-AP (Figure 12) flow 2 does not share the available bandwidth with flow 1 and results in low fairness. In Figure 14 we show the aggregate throughput for all TCP versions. It is clear that M-TCP at x = 50 can reach the other TCP throughput or better with highest fairness.

Our approach proves a high responsive to mobility. In Scenario 5 (Figure 13), While standard TCP flow 1 fall into congestion problems at second 90, it *cannot* recover when hidden terminal exist at second ≈130. M-TCP continues transmission producing reasonable throughput with high fairness. We only plot M-TCP and standard TCP curves since the other TCPs have the same behavior. M-TCP clearly provides a comprehensive solution to the problem of hidden terminals in an ad hoc chain, cross and random networks, since it allows the two flows to share the bandwidth in a fair manner.

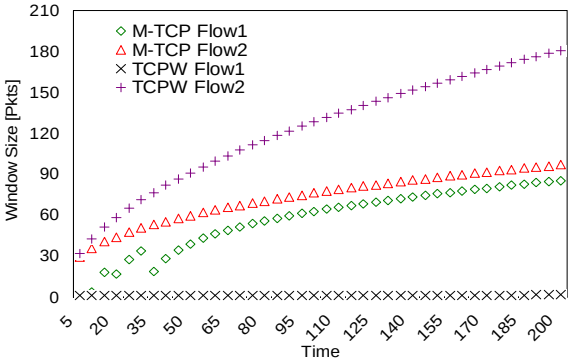

Fig. 9 M-TCP and TCPW flows, scenario 2

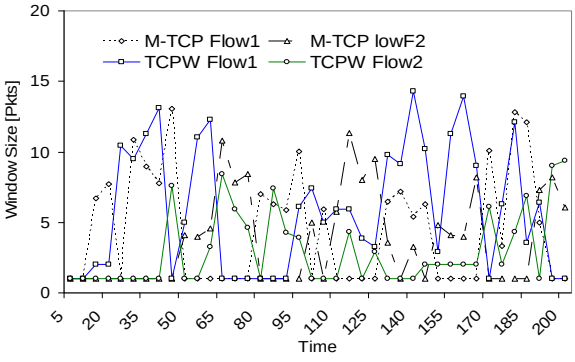

Fig. 10 M-TCP and TCPW flows, scenario 4

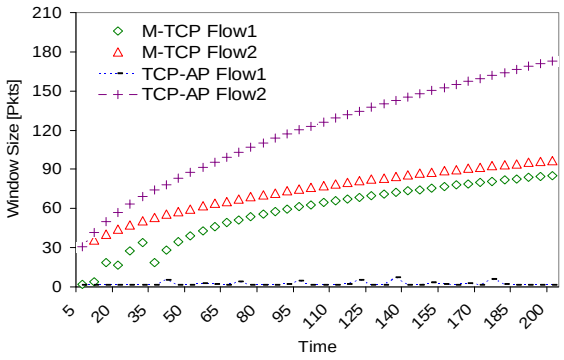

Fig. 11 M-TCP and TCP-AP Flows, scenario 2

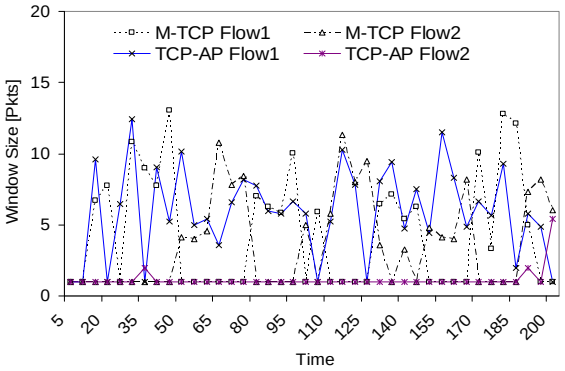

Fig. 12 M-TCP and TCP-AP flow, scenario 4

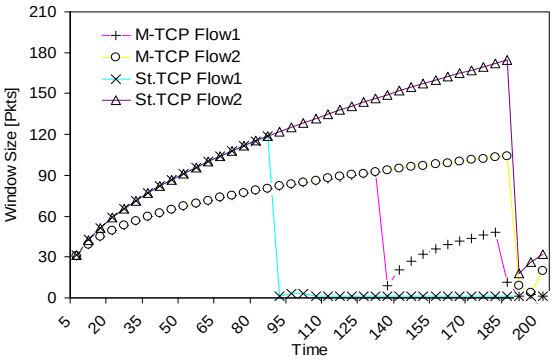

Fig. 13 M-TCP and standard TCP, scenario 5

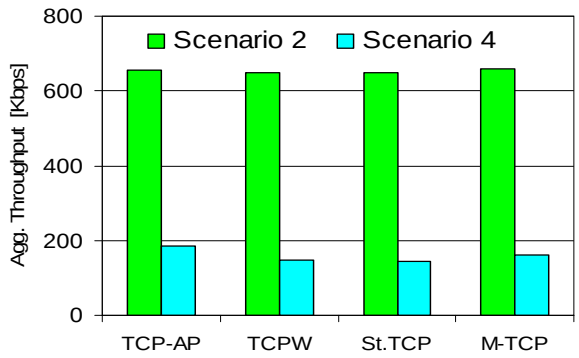

Fig.14 Throughput of M-TCP and other TCP variants when hidden terminal exist.

6 Conclusion

We studied the impact of a shared medium on TCP performance. We have proposed M-TCP, a simple algorithm to improve the fairness of flows sharing a single available bandwidth channel. Our simulations of various network topologies show that over ad hoc wireless networks, the fairness of shared flows improves significantly if they adopt the packet transmission interval that achieves the highest possible spatial channel reuse. In all scenarios it is possible to achieve a high level of fairness while maintaining reasonable throughput and minimizing loss.

We also studied the throughput as a function of the transmission interval and the TCP algorithm used. We found that M-TCP can improve throughput by 1.4% in a chain network and by 5% in cross network, with respect to standard TCP and Westwood. Our approach also proves high responsiveness towards mobility producing a high fairness and reasonable throughput.

References

[1] Z. Fu, P. Zerfos, H, Luo, S. Lu, L. Zhang, and M. Gerla. The impact of multihop wireless channels on TCP throughput and loss. *INFOCOM'03*, 2003

[2] J. Liu and S. Singh. ATCP: TCP for mobile ad hoc networks, *IEEE J--SAC*, Vol. 19, No. 7, 2001, pp 1300-1315.

[3] S. ElRakabawy A. Klemm and C. Lindemann. TCP with Adaptive Pacing for Multihop Wireless Networks, ACM *MobiHoc 2005*, 2005

[4] http://www.isi.edu/nsnam/

[5] Saverio Mascolo, Claudio Casetti, Mario Gerla, M. Y. Sanadidi and Ren Wang. TCP Westwood: Bandwidth estimation for enhanced transport over wireless links. *Mobile Computing and Networking*, 2001 pp 287-297

[6] A. Singh P. Ramanathan and B. Van Veen Spatial reuse through adaptive interference cancellation in multi-antenna wireless networks. *In Proc. GLOBECOM'05*, Dec 2005[7] C. L. Fullmer and J.J. Garcia-Luna-Aceves Solutions to Hidden Terminal Problems in Wireless Networks. *In Proc. ACM SIGCOMM'97*, Sep.1997

[7] C. L. Fullmer and J.J. Garcia-Luna-Aceves Solutions to Hidden Terminal Problems in Wireless Networks. *In Proc. ACM SIGCOMM'97*, Sep.1997

[8] 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.

[9] Zhifei Li, S. Nandi and A.K. Gupta Achieving MAC Fairness in Wireless Ad-hoc Networks using Adaptive Transmission Control, *In proc. ISCC 2004*, Vol 1, No 28, July 2004, pp 176-181.

[10] C. Perkins and S. Das Ad hoc On-Demand Distance Vector Routing. *RFC 3561*, July 2003. http://www.rfc-archive.org/getrfc.php?rfc=3561

[11] R. Jain D. Chiu and W. Hawe, A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Systems, *DEC Technical Report DEC TR-301*,1984.

[12] Thomas D. Dyer Rajendra V. Boppana. A comparison of TCP performance over three routing protocols for mobile ad hoc networks. *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking and computing*, October 2001.

[13] K. Chandran S. Raghunathan S. Venkatesan and R. Prakash. A feedaback based scheme for improving tcp performance in ad-hoc wireless networks. *IEEE personal Communications*, 8(1):34-39, February 2001.

[14] Xin Yu. Improving tcp performance over mobile ad hoc networks by exploiting crosslayer information awareness. *Proceedings of the 10th annual international conference on Mobile computing and networking*, September 2004.

[15] G. Holland and N. Vaidya. Analysis of tcp performance over mobile ad hoc networks. *MobiCom'99: Proceedings of the 5th annual ACM/IEEE*, pages 219-230 USA, August 1999.