Wireless TCP in Unfair Situation: Performance Degradation and Improvement

Md. Shohidul Islam, W.H. Sadid, M. A Kashem, M. A Rahman, M.N Islam

Abstract— **Traditional TCP implementations are tuned to work well over wired networks. A packet loss is occurred in a wired network mainly due to network congestion. On the other hand in a wireless link packet losses are caused mainly due to bit errors resulted from noise, interference, and various kinds of fadings. TCP performance in these environments is impacted by three path characteristics, not normally present in wired environments: high bandwidth delay product, packet losses due to corruption and bandwidth asymmetry. Wireless TCP has no idea whether a packet loss is caused by congestion or bit error. TCP assumes loss is caused by congestion and turns on its congestion control algorithms to slow down the amount of data it transmits as well as adopts retransmission policy. Invoking congestion control for bit errors in wireless channel reduces TCP throughput drastically. We propose an empirical architecture to recover these bit errors at Data Link Layer dynamically before entering the frame into buffer which reduces the number of retransmission at TCP level and thus preserves time to send more data, as our simulation results show. Consequently overall throughput can be increased where retransmission technique is almost 100% ineffective for unfair noisy channel.**

Index Terms— **Frame, AFEC, Fading, Interference.**

I. Introduction

In general TCP, packet transmission follows buffered model in which one buffer is for sender and other for receiver [6, 7] that accepts packets according to queue order. Our proposed architecture includes a bypass mechanism which acts as error detection and correction scheme at Data Link layer of transmission. Incase of TCP, for each packet (fair or corrupted) reached in receiver, acknowledgement is sent to the sender [7]. So basic traveling period spent for an i-th packet is given by $t(i) = 2 * \partial_i$, where ∂_i is one way quantum needed for i-th packet or acknowledgement to travel from source to destination or vice-versa.

Manuscript received on October 30, 2008. This work was supported in part by the IAENG Hong Kong and Department of Computer Science & Engineering, Dhaka University of Engineering & Technology, Gazipur-1700, Bangladesh.

M. Shohidul Islam, M.A. Kashem and M.N Islam are with the Department of Computer Science & Engineering DUET, Gazipur-1700, Bangladesh. (e-mail: shohidulcse@duet.ac.bd, drkashemll@duet.ac.bd, nazrul_ruet@yahoo.com). W.H Sadid is with the Concordia University, Canada. (email:whsadid@yahoo.com). M. A Rahman is with the DEBII institute, Curtin University of Technology, GPO Box U1987, WA 6845 Perth, Australia. (email: M.Rahman@cbs.curtin.edu.au).

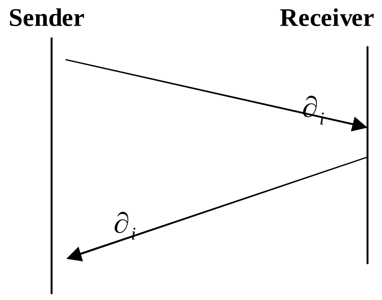

Fig. 1 Packet and Acknowledgement Transmission

The principle of TCP is to retransmit a packet several times if it is not successfully reached to receiver. This *retransmission technique* is ineffective incase of erroneous unfair wireless channel where packet will be always corrupted before reaching destination and $2 * \partial_i$ time will be spent ideally. To get ride of this situation we have proposed the model that yields desired solution as our Network Simulator (NS-2) simulation depicts.

II. Mathematical Basis of Proposed Model

Consider N_{pd} is number of corrupted packets, $t_r(N_{pd})$ and $t_c(N_{pd})$ indicates retransmission time and correction time of N_{pd} packets respectively. Theoretically

$$t_r(N_{pd}) > t_c(N_{pd})$$

$$t_r(N_{pd}) = 2 * \sum_{i=1}^{N_{pd}} \partial_i$$

$$t_c(N_{pd}) = N_{pd} * T_c$$

$$T_{save} = t_r(N_{pd}) - t_c(N_{pd})$$

$$T_{NEW} = T_{OLD} + T_{save}$$

$$N_{AFEC} = N_{T_{NEW}}$$

$$\because T_{NEW} > T_{OLD}$$

$$\therefore N_{AFEC} > N_{GTCP}$$

Where T_{save} is saved time, obtained by correcting the packets in place of retransmission. Thus it provides much time interval within which more and more packets are sent. N_{AFEC} refers the number of packets that can be successfully sent in our Adaptive Forward Error Correction (AFEC) model and N_{GTCP} refers the packets for general TCP with traditional model. Mathematical relationship shows that our proposed architecture provides more packets than that of traditional model within same duration which proves its effectiveness. Following experimental results express that TCP performance will be improved regarding to other perspectives also by using this model.

III. TCP Performance Improvement

Our attempt aimed to improve two undesired features (i.e. unwise drop of congestion window and several times re-transmission of same packet) of existing wireless TCP by one means- Packet Correction at destination end before entering it into the buffer which yields optimal buffer utilization and preserves time to send more data. Many error detection and correction codes such as Hamming code, Reed Solomon (RS) code, CRC e.t.c are in existence but we prefer Hamming code for its lower order complexity (*O(n)* for n bit packet) whose principle would be shortly mentioned later.

IV. Policy of Hamming Code

The function H (m+r, m) indicates that if original data block has 'm' bits, it is enlarged to 'm+r' by adding 'r' redundant checksum bits where 'r' is calculated from the following inequality [17].

$$2^r \geq m + r + 1.$$

A. Position of Redundant Bits

Position of the checksum bits ($R_1, R_2, \ldots R_r$) in the packet are determined by $R_{i+1} = 2^i$ where i=0, 1, 2 …………(r-1). For instance seven (7) bit data bock needs four (4) check sums as below.

B. Values of Redundant Bits

All the 'r' bit binary numbers from '1' to 'm+r' (here 1 to 11 for 4 redundant bits) have to be considered shown in table I.

Table I. Cell selection to calculate redundant bits

DECIMAL	BINARY			
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
11	1	0	1	1

For each binary sub column from LSB to MSB, we have to consider the cell containing '1'. Then the corresponding decimal numbers indicate the original data bit numbers on which X-OR (exclusive OR) operation has to be performed to get 'R' values. Thus we get:

$$R_1 = H_3 \oplus H_5 \oplus H_7 \oplus H_9 \oplus H_{11}$$
$$R_2 = H_3 \oplus H_6 \oplus H_7 \oplus H_{10} \oplus H_{11}$$
$$R_3 = H_5 \oplus H_6 \oplus H_7$$
$$R_4 = H_9 \oplus H_{10} \oplus H_{11}$$

where any H_i indicates i th bit of hamming code.

C. Error Detection and Correction

At the receiver end, all check bits

$$C_1 = H_1 \oplus H_3 \oplus H_5 \oplus H_7 \oplus H_9 \oplus H_{11}$$
$$C_2 = H_2 \oplus H_3 \oplus H_6 \oplus H_7 \oplus H_{10} \oplus H_{11}$$
$$C_3 = H_4 \oplus H_5 \oplus H_6 \oplus H_7 \text{ and}$$
$$C_4 = H_8 \oplus H_9 \oplus H_{10} \oplus H_{11}$$

are re-calculated. If ($C_4\ C_3\ C_2\ C_1$) represent 'ZERO' then no error occurred, otherwise error occurred and the non-zero value indicates corrupted bit position in the received packet. Simply perform negation ('NOT') operation on the value of corrupted bit position. That is if D_{OLD} be corrupted bit then after error correction we will get $D_{NEW} = \overline{D_{OLD}}$.

V. Effect of AFEC

A. Packet Vs Time

Table II and Figure 4 show that during certain simulation times, amount of fair packets at receiver buffer were higher for proposed TCP than that of traditional TCP. In noisy channel if any packet became corrupted, TCP retransmit its fresh copy several times which unfruitfully consume more CPU cycles, channel bandwidth and other resources. Everything can de reduced by packet correction at destination because corrected packet does not need re-transmission, as a result we can obtain relatively more and more data within same time interval.

Table II. Packets for 'Traditional' and 'Proposed' TCP

Simulation Time (Sec)	Packet received for	
	Traditional TCP	Proposed TCP
0	0	0
5	403	450
10	813	903
20	1611	1789
30	2416	2684
40	3268	3631
50	4035	4483
60	4909	5454

Figure 2 expresses nature of characteristics curves that corresponds table II.

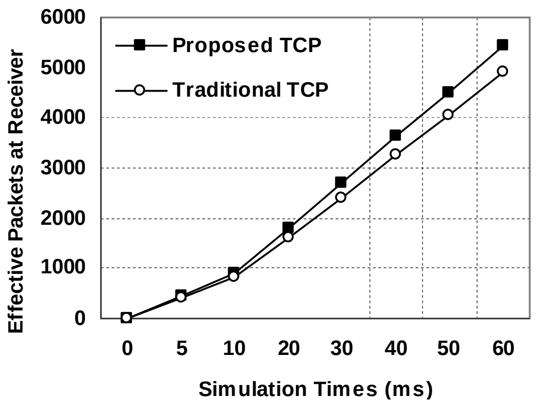

Fig. 2 Graph of packet vs time

B. Buffer Utilization

T_s and A_s refers Total size (predefined) and Active buffer size. Corrupted packets are never included in the buffer so a large volume of buffer remains idle for general TCP, which indicates poor resource utilization. But their instantaneous correction solves the incident wisely. Proportion of idle buffer we obtain from the equation 1.

$$Percentage\_of\_Idle\_buffer = \frac{(T_s - A_s)}{T_s} * 100\% \quad \textbf{..........(1)}$$

As time passes, active size of buffer begins to increase thus percentage of idle buffer decreases. Corrupted packets are never included into buffer, so a large portion of it remains idle for noisy channel as found in the above data table incase of 'General TCP'. But instantaneous correction of packets (which is our target) at this point can fruitfully improve such undeserving behavior of wireless TCP. Incase of 'Proposed TCP', the proportion of idle buffer is much less than that of 'General TCP' which implies higher degree buffer utilization.

Table III. Proportion of Idle Buffer for 'Traditional' and 'Proposed' TCP

Simulation time (Sec)	% of Idle Buffer (Receiver)	
	Traditional TCP	Proposed TCP
0	100%	100%
5	93.28%	92.41%
10	86.45%	84.95%
20	73.15%	70.18%
30	59.73%	55.26%
40	45.53%	39.48%
50	32.75%	25.28%
60	18.18%	9.10%

VI. Conclusion

The proposed model improves wireless TCP performance in unfair situation when probability of packet loss booms almost 100% , ensures higher throughput as well as optimal buffer utilization , reduces number of idle re-transmission and unwise drop of congestion window effectively in an efficient way. Testing the performance of this model by applying Reed Solomon or other upcoming latest error detection and correction code will be an appreciating future attempt.

References

[1] NS, The network simulator-ns-2.27.*URL:http://www.isi.edu/nsnam/ns*

[2] Benyuan Liu, DennisL.Goeckel and Don Towsley, "TCP-Cognizant Adaptive Forward Error Correction in Wireless Networks", Department of Computer Science, University of Massachusetts, Department of Electrical and Computer Engineering, University of Massachusetts.

[3] Paul Meeneghan and Declan Delaney, "An Introduction to NS, Nam and OTcl scripting" National University of Ireland.

[4] Jae Chung and Mark Claypool, "NS by Example".

[5] "NS Simulator for beginners", Lecture notes, 2003-2004, Univ. de Los Andes, Merida, Venezuela and ESSI, Sophia-Antipolis, France, December 4, 2003.

[6] "ns Tutorial", *http://www.isi.edu/nsnam/ns/tutorial/nsindex.html*

[7] "The ns Manual", The VINT Project, A Collaboration between researchers at UC Berkeley, LBL, USC/ISI, and Xerox PARC. December 13, 2003.

[8] Minna Kaisa and Juonolainen, "Forward Error Correction in INSTANCE",UNIVERSITY OF OSLO, Department of Informatics.

[9] Henrik Lindquist and Gunnar Karlsson, "TCP with end-to-end FEC", In Proceedings of International Zurich Seminar on Communications, pages 152 – 155, Zurich, Switzerland, February 2004.

[10] Hasegawa, Kenji Kurata and Masayuki Murata "Analysis and Improvement of Fairness between TCP Reno and Vegas for Deployment of TCP Vegas to the Internet", Graduate School of Engineering Science, Osaka University.

[11] Michele Zorzi, A. Chockalingam, and Ramesh R. Rao, "Throughput Analysis of TCP on Channels with Memory", Senior Member, IEEE.

[12] Claudio Casetti, MarioGerla, Scott Seongwook Lee, Saverio Mascolo and Medy Sanadidi, "TCP WITH FASTER RECOVERY", Computer Science Department, University of California, LosAngeles, USA.

[13] Hari Balakrishnan, Venkata N. Padmanabhan, Srinivasan Seshan, and Randy H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links", Student Member, IEEE.

[14] HALA ELAARAG, "Improving TCP Performance over Mobile Networks", Stetson University.

[15] Ling-Jyh Chen, Tony Sun, M. Y. Sanadidi, Mario Gerla, "Improving Wireless Link Throughput via Interleaved FEC", UCLA Computer Science Department, Los Angeles, CA 90095, USA {cclljj, tonysun, medy, gerla}@cs.ucla.edu.

[16] V. Jacobson, "Congestion Avoidance and Control", Computer Communication Review, vol. 18, no. 4, pp. 314--329, Aug. 1988. *ftp://ftp.ee.lbl.gov/papers/conga-void.ps.Z.*

[17] Behrouz A. Forouzan, "Data Communications and Networking", Mc Graw Hill, 3rd edition.

[18] T. Kim, S. Lu, V. Bharghavan, "Improving congestion control performance through loss differentiation", Proceedings Eight International Conference on Computer Communications and Networks IEEE. 1999, pp.412—18.