Techniques for Enhancing the Performance of TCP in Wireless Networks

MD. Sirajuddin, Ch. Rupa and A. Prasad

Abstract TCP is a connection-oriented and reliable transport layer protocol. Currently, it is most widely used protocol in the Internet. All the reliable Internet applications make use of this protocol. This protocol works well in the wired network, but it does not produce satisfactory results if it is used in the wireless networks. Usage of TCP in wireless networks leads to performance degradation, because it considers all the packets losses as due to congestion and reduces the packet sending rate, and at the same time, it diminishes the network throughput. This feature is suitable for wired networks in which packet loss mainly occurs due to congestion. However, this is inappropriate in wireless networks where packet loss occurs due to signals fading, high bit error rate, hand-off, etc. It misinterprets all packet losses as due to congestion and reduces its congestion window. This misinterpretation of packet loss mainly decreases the throughput. This paper depicts the performance of all the efficient mechanisms which have been developed to improve the performance of TCP.

Keywords Wireless networks · Congestion control · Improved mechanisms · RTT

MD. Sirajuddin (✉)
Department of C.S.E, S.M.C.E, Guntur, Andhra Pradesh, India
e-mail: siraj.cs@gmail.com

Ch. Rupa
Department of C.S.E, VRSEC, Vijayawada, Andhra Pradesh, India
e-mail: rupamtech@gmail.com

A. Prasad
Department of C.S, Vikrama Simhapuri University, Nellore, Andhra Pradesh, India
e-mail: prasadjkc@yahoo.co.in

L.P. Suresh et al. (eds.), *Artificial Intelligence and Evolutionary Algorithms in Engineering Systems*, Advances in Intelligent Systems and Computing 324,
DOI 10.1007/978-81-322-2126-5_18

1 Introduction

TCP [1] is most widely used protocol in the Internet. All the Internet applications which provide reliability make use of TCP. When it came into an existence all the networks were made up of wired links. It was designed for wired networks. In such networks, the packet loss triggers TCP congestion control algorithm at sender and hence reduces the congestion window. The congestion control mechanism of TCP is very effective in the wired networks because the packet loss rate is very low. In such networks, packet loss occurs mainly due to the congestion. In effect, the TCP congestion algorithm cannot differentiate the packet losses caused by congestion from the one caused by error. Once a loss occurs, TCP considers it as a loss occurred due to congestion and reduces its congestion window. This leads to performance degradation and underutilization of channel bandwidth. This behavior is acceptable in wired networks because 99 % packet loss occurs due to congestion. However, this behavior is not acceptable in wireless networks because packet loss in such networks cannot be designated as due to congestion; there are also many reasons for packet losses such as interferences, signal fading, hand-off, and other radio effects.

TCP is unable to differentiate between losses due to congestion and due to corruption; it became the main cause of performance degradation of TCP on wireless links. This problem is popularly known as the TCP performance problem over wireless network [2]. In this paper, we study the performance of various techniques to improve the performance of TCP over wireless links. This paper will be useful for analyzing all the existing methods and for implementing innovative congestion control schemes.

The rest of paper is structured as follows. Section 2 describes the performance issues of TCP in wireless networks. Section 3 represents the existing techniques which addresses the performance issues. Section 4 consists of proposed solution. Section 5 represents the conclusion.

2 Obstacles in TCP

TCP is responsible for end-to-end delivery of messages. In wireless network, the performance of TCP is more important, because in such networks possibility of error rate is very high. For reliable data transmission, TCP uses congestion window. This congestion window specifies the number of packets that can be sent by the sender without worrying about acknowledgment. When packet loss occurs, the sender TCP decreases the congestion window directly without investigating the reason for packet loss. There are two reasons for packet loss, first due to congestion and second is due to link failure or noise in wireless links. If packet loss occurs due to link errors, the reducing congestion window leads to the reduction in data transmission rate. This feature affects the throughput of the network. The congestion window should be reduced only in the presence of congestion. So, we need to design an intelligent congestion control scheme which will identify the reason for packet loss and then decide whether to decrease the congestion window or not.

2.1 TCP Issues in Wireless Netwoks

For reliable and effective data transmission, the performance of the TCP needs to be improved. The conventional TCP which was developed for wired networks is to be modified to support existing wireless network features with minimum overhead and able to address all the below listed issues.

- How to detect congestion and reduce data rate?
- How to detect multiple packet loss within the same window?
- Time required to detect each packet loss should be minimized.
- How to determine the condition of wireless link?
- TCP should only retransmit the lost packet caused by bit error and do not reduce the size of congestion window.
- All the transmission error caused by bit error should be transparent to the sender and it can be resumed quickly.

3 Existing Approaches

There are many approaches that have been developed to solve this misinterpretation of packet loss in TCP. Whenever packet loss occurs, TCP misinterprets that this packet loss was due to congestion and reduces its congestion window. In this section, we have presented various schemes that solve above-mentioned problems. Each existing approach is explained briefly along with its strengths and weaknesses by the following ways.

3.1 Fast Retransmission and Fast Recovery

In order to detect congestion, TCP Tahoe [3] makes use of slow start, congestion avoidance, and fast retransmission. This algorithm gives better result initially. As soon as the packet is lost, it reduces the data rate. This algorithm reduces congestion window without identifying the reason for packet loss. It has a drawback of drastic reduction in congestion window.

This limitation of TCP Tahoe is overcome by TCP Reno [4]. It based on TCP Tahoe, but uses the concept of duplicate acknowledgments to trigger fast retransmission. When TCP Reno receives 3 duplicate acknowledgments, it enters fast recovery. In fast recovery, congestion window is set to half the value of current window. But, this technique cannot detect multiple packet loss within the same window.

The limitation of TCP Reno is overcome by TCP New-Reno [5, 6]. In this protocol, fast recovery scheme is modified to deal with multiple packet losses in a single window of data. It is inefficient in terms of bandwidth. TCP new-Reno suffers from the fact that it takes one RTT to detect each packet loss. All these three techniques will address three issues of Sect. 2, but misinterpretation of packet loss still exists.

3.2 Determining the Cause of Packet Loss

The cause of packet loss can be determined by considering the network parameters. Xiao et al. [7] proposed a mechanism in which change trend of RTT is used to determine the cause of packet loss and the status of the network. When the RTT between consecutive packets increases continuously, then it signals the occurrence of congestion. In this approach, certain parameters are used to detect the condition to link that are as follows: $\mathrm{t_i}$ denotes the time of the packet received, Receive Packet Time Interval (RPTI) denotes the interval time of two consecutive received packets of the receiver, and λ denotes the difference between two consecutive RPTI. That is,

$$\mathrm{RPTI}_i = t_i - t_{i-1};\ \mathrm{RPTI}_{i-1} = t_{i-1} - t_{i-2;}\ \lambda = \ \mathrm{RPTI}_i - \ \mathrm{RPTI}_{\mathrm{i}-1}$$

Wireless link condition can be determined according to the λ on the receiver. This technique is compatible with IPSec.

3.3 Congestion Control Mechanism for Wireless Networks and Detection of Multiple Packet Loss

An alternative congestion control mechanism which is specially designed for wireless networks is TCP Westwood [8]. It uses TCP Reno operation with exponential growth during slow start and linear growth during congestion avoidance phase. Based on the packet sizes and RTT estimates, it uses a series of equations to estimate the bandwidth usage of the link. When a packet is lost, the window is reset to the bandwidth estimate rather than reducing it to half. It is having a drawback that sometimes it overestimates the available bandwidth which affects the throughput of the network.

To detect multiple packet loss within the same window of data, Selective Acknowledgements scheme (SACK) [6] was developed. SACK scheme supports recovery of multiple packet loss within the same RTT window. SACK can specify which blocks of data, following the loss have been successfully received. By informing the sender which packets have been received, and which packets must be resent, multiple lost packets can be recovered in one RTT. SACK is proven to be beneficial [5] for error recovery in wireless networks.

To improve the performance, modified version of SACK TCP was proposed by Metha and Vithalani which is called as SACK_OK [9]. They used a flag SACK_OK to differentiate between random loss and congestion loss. It avoids the reduction of congestion window when the random loss is identified. This mechanism uses distance as another parameter to handle the fast retransmission. This technique does not require changes on receiver side and in header also. The process of modified SACK TCP is shown in Fig. 1.

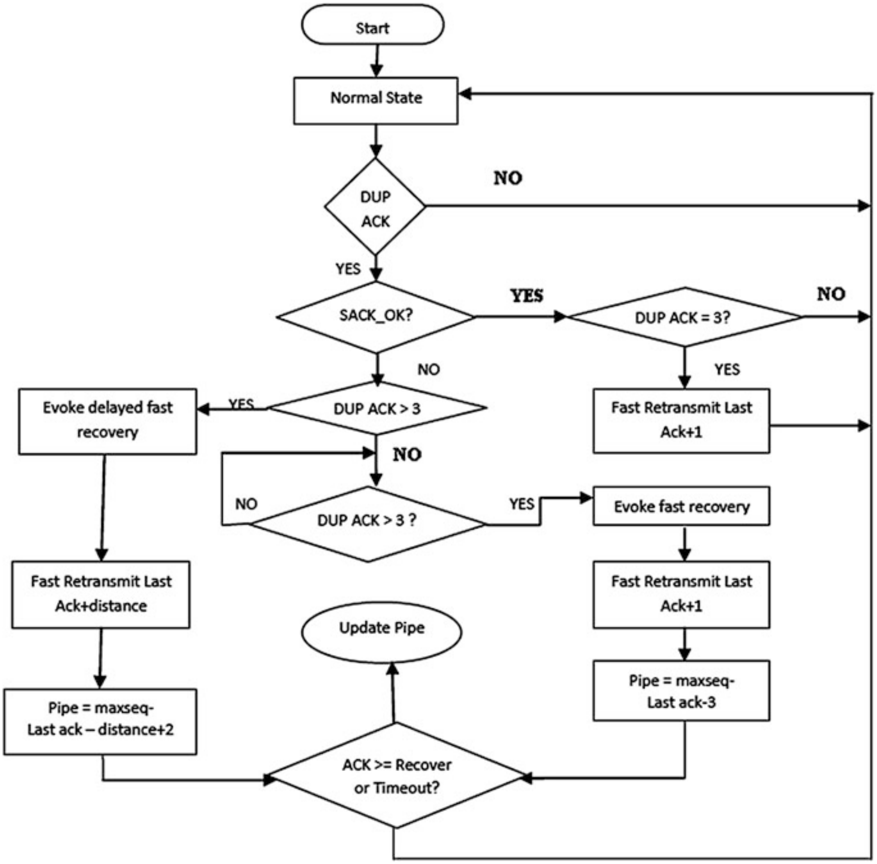
Start
Normal State
DUP ACK
NO
YES
SACK_OK?
YES
DUP ACK = 3?
NO
NO
YES
Evoke delayed fast recovery
YES
DUP ACK > 3
Fast Retransmit Last Ack+1
NO
NO
DUP ACK > 3 ?
YES
Evoke fast recovery
Fast Retransmit Last Ack+distance
Fast Retransmit Last Ack+1
Update Pipe
Pipe = maxseq-Last ack-3
Pipe = maxseq-Last ack – distance+2
ACK >= Recover or Timeout?

Fig. 1 Flow chart of modified SACK TCP

3.4 Explicit Congestion Notification and Usage of PEP

It is also possible to use certain bits in the header to signal the occurrence of congestion. Explicit congestion notification [10] uses two bits in the IP header and two bits in the TCP header to record the status of the network.

To improve the throughput of TCP, performance enhancing proxy (PEP) can also be used. PEP provides reliability in wireless networks with minimum overhead [11–14]. Split TCP and Snoop are commonly used PEPs. These two approaches suffer from the number of problems which are listed below.

- This scheme breaks TCP end-to-end semantics.
- Path failure between PEP and receiver TCP will lead to unexpected problems.
- Does not support route changes between the sender TCP and the PEP.
- Many security protocols are not compatible with these approaches.
- Snoop proxy does not specify what should be done if the retransmitted packet is also lost.
- Cannot able to hide TCP losses due to interoperability problems.

Above-stated limitations are overcome by innovative proactive distributed TCP proxy called D-proxy [2, 8]. D-proxy is distributed because it uses a proxy either side of the wireless link. It is proactive. The basic concept of D-proxy is relatively simple, but the implementation is complex. D-proxy is able to maintain and recover losses in very high loss situations. It can perform this task using negative acknowledgments. The key benefit is that D-proxy is a negatively acknowledging proxy; sending messages only when a packet is missing. This technique is dependent on D-proxy. Problem occurs when the D-proxy fails.

3.5 Using Reserved Bits of TCP Header and SNR Ratio

Another congestion control scheme was proposed by Bassil [2]. It uses one of the reserved bits of the TCP header and SNR ratio to detect the reliability of the link and decide whether to reduce the packet burst or to retransmit the lost packet. The reserved bit (RB) specifies the type of the link over which the TCP connection is established. For wired link, the RB = 0 and for wireless, RB = 1. In wired mode, any timeout is considered a congestion loss. If packet loss occurs in wireless link, then SNR ratio of the link is measured. Based on SNR value, the decision is made whether to reduce the congestion window or to retransmit the packet. This process is shown in Fig. 2. This scheme maintains true end-to-end semantics, without involving any proxies between the sender and the receiver. This feature eliminates extra processing overhead and the need for extra buffer space. It does not require modifying the source and receiver code.

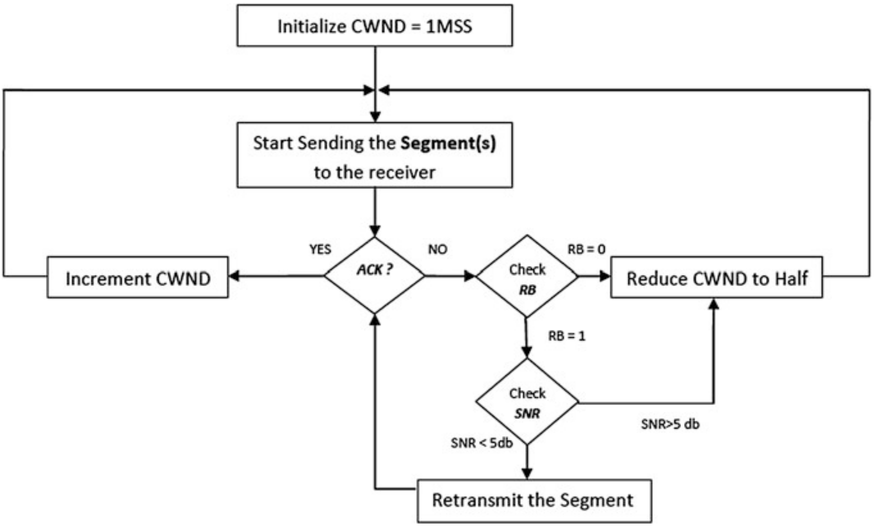

Fig. 2 Flow chart for detecting the cause for packet loss

3.6 *WiTracer*

Recently proposed approach to improve the performance of TCP is WiTracer [15]. It addresses various issues of TCP Reno and gives solutions for them. TCP Reno suffers from various problems such as Hungry RTO, RTX-DATA Loss, and spurious DATA. WiTracer solves all these problems by using congestion identifier and opportunistic-recovery scheme. Congestion identifier checks for each duplicate acknowledgment packet and implement local recovery rate, while opportunistic recovery provides a faster recovery rate of DupAck and overcomes several TCP drawbacks. WiTracer can be installed on mobile devices for various applications.

4 Proposed Solution

The proposed scheme uses certain network parameters to determine the status of the network and the reason for the packet loss. The performance of TCP can be improved by considering the parameters such as RTT, throughput, and by setting the timers carefully. By setting the timers carefully, we can reduce the delay in wireless networks.

The best technique to solve TCP congestion control problem is to use RTT and throughput. In proposed scheme, for every packet, TCP records the RTT and it also estimates the throughput. By using these two parameters, the TCP can determine the status of the network and can able to detect the real cause of packet loss.

The RTT between consecutive packets is compared. Increasing RTT and decreasing throughput signals the occurrence of congestion. Whenever the packet is lost, the TCP estimates the throughput and also compares the latest RTT with the previous RTT's in a record and also with the threshold value let it be $\mathrm{RTT}_{\mathrm{Thresh}}$. If the latest RTT reaches the $\mathrm{RTT}_{\mathrm{Thresh}}$, then the TCP considers the packet loss as due to congestion and hence reduces the congestion window based on the capacity of the network. If RTT between packets is uniform and less than the $\mathrm{RTT}_{\mathrm{Thresh}}$, then the packet loss is interpreted as due to link error. In this case, TCP only retransmits the lost packet without reducing the congestion window. But, when the packet is retransmitted, the RTO [16] value is doubled. For efficiency, the RTO value should be doubled only when the packet loss is due to congestion, otherwise it will be same. The following algorithm can be used to set the RTO value whenever the packet needs to retransmitted.

4.1 Algorithm

When timeout occurs, throughput and RTT values are considered to know the status of congestion. If occurrence of congestion is confirmed, then the RTO value is doubled and the lost packet is retransmitted, otherwise RTO value is kept same. This process is shown by the following steps.

```
Step 1: When timeout occurs
Step 2: if(Throughput * RTT > congestion_window)
Step 3: then no congestion
Step 4: Retransmit the lost packet with the same RTO value.
Step 5: else RTO = 2 * RTO
Step 6: End-if
```

By considering above-stated parameters, the TCP congestion control problem can be solved. Alternatively, we can also use cross-layer approach in which the below layer will provide the information to the above layer. Generally, the network layer monitors the operation of subnet, and it also controls the congestion; it would be better to provide the status of the network by the network layer to the transport layer so that the transport layer can know the real cause of packet loss.

5 Conclusion

In this paper, we have explained various causes that results in TCP performance degradation and proposed some solutions to this problem. This paper presents various techniques that can be used for improving the performance of TCP both in wired as well as in wireless networks. This paper would be helpful for the researchers to better analyze the TCP performance problems and to invent better solutions for it.

References

1. J. Postel, Transmission control protocol specification. RFC 793 (1981)
2. Y. Bassil, TCP congestion control scheme for wireless networks based on tcp reserved field and snr ratio. IJRRIS **2**(2) (2012)
3. M. Tayade, S. Sharma, Performance improvement of TCP in MANET. IJEST **3**(3) (2011)
4. B. Moraru, F. Copaciu, G. Lazar, V. Dobrota, Practical analysis of TCP implementations: Tahoe, Reno, New Reno
5. S. Floyd, T. Henderson, A. Gurtov, The new reno modification to TCP's fast recovery algorithm. RFC 3782 (2004)
6. M. Mathis, J. Mahadevi, S. Floyd, A. Romanw, TCP selective acknowledgement options (1996)
7. L. Xiao, Z. Li, N. Zhao, A TCP performance improved mechanism for wireless network. UIC-ATC. In *Symposia and Workshops on Ubiquitous, Autonomic and Trusted Computing*. IEEE, pp. 212–216 (2009)
8. D. Murray, T. Koziniec, M. Dixon, D-Proxy: reliability in wireless networks. In *2010 16th Asia Pacific Conference on Communications (APCC)*, pp. 129–134
9. R.D. Metha, C.H. Vithalani (2012) Distinguishing congestion loss from random loss on wireless errorneous links to improve performance of wireless TCP-SACK. In *CSNT '12 Proceedings of the 2012 International Conference on Communication Systems and Network Technologies*. IEEE, pp. 383–387 (2012)
10. K. Ramakrishnan, S. Floyd, A proposal to add explicit congestion notification to IP. Internet Draft (1999)
11. J. Border, M. Kojo, J. Griner, G. Montenegro, Z. Shelby, Performance enhancing proxies intended to mitigate link-related degradations. IETF RFC 3135 (2001)
12. A. Bakre, B.R. Badrinath, I-TCP: indirect TCP for mobile hosts. In *15th International Conference on Distributed Computing Systems. ICDCS'95 Proceedings of the 15th International Conference on Distributed Computing Systems*. IEEE, pp. 136–143 (1995)
13. H. Balakrishnan, S. Seshan, E. Amir, R.H. Katz, Improving TCP/IP performance over wireless networks. In *MobiCom'95 Proceedings of the 1st annual international conference on Mobile computing and networking*, ACM, pp. 2–11 (1995)
14. B. Francis, V. Narsimhan, A. Nayak, I. Stojmenovic, Techniques for enhancing TCP performance in wireless networks. In *2012 32nd International Conference on Distributed Computing Systems Workshops*, pp. 222–230 (2012)
15. C. Hu, X. Yang, M. Fan, P. Zhao, WiTracer: a novel solution to improve TCP Performance over Wireless Network. In *9th IEEE International Wireless Communications and Mobile Computing Conference*. IEEE, pp. 450–455 (2013)
16. H. Touati, I. Lengliz, F. Kamoun, TCP adaptive RTO to improve TCP performance in mobile ad-hoc networks, pp. 176–180 (2007)