EURASIP Journal on Wireless Communications and Networking
a SpringerOpen Journal

REVIEW Open Access

TCP performance in multi-hop wireless ad hoc networks: challenges and solution

Ammar Mohammed Al-Jubari[1*], Mohamed Othman[1], Borhanuddin Mohd Ali[2] and Nor Asilah Wati Abdul Hamid[1]

Abstract

Transmission control protocol (TCP) performance over multi-hop wireless networks is currently attracting considerable interest from the research community. The characteristics of multi-hop wireless networks, such as mobility, link layer contention, high bit error rate, asymmetric path, network partition, hidden exposed nodes and dynamic routing, do not fit the requirements of TCP for a good reliable data delivery. Here we want to provide an overview of the research progress in applying TCP algorithms to the problem of multi-hop conditions and characteristics. The scope of this review will encompass core methods and protocols of TCP over multi-hop networks, including cross-layer, network layer protocols and medium access control (MAC) layer protocols. The research contributions in each field are systematically summarized and compared, allowing us to clearly define existing research challenges, and to highlight promising new research directions. The findings of this review should provide useful insight into current multi-hop networks literature and be a good source for anyone who is interested in "TCP in wireless" approaches or related fields.

Keywords: TCP, multi-hop wireless network, survey

1 Introduction

The rapid advancement in portable computing platforms and wireless communication technology has led to making it an integral part of the Internet. As the Internet users' requirements of flexibility and mobility increase, multi-hop wireless networks have become the best solution to satisfy these requirements. These networks are complex distributed systems that consist of wireless mobile or static nodes. Compared to traditional infrastructure wireless networks, multi-hop ad hoc networks have many features: dynamic self-organization, self-configuration, free movement, high scalability, low deployment cost, robustness and easy maintenance. These features give ad hoc networks the ability to play a leading role in the next generation of wireless networks. Recently, the introduction of new standards such as Bluetooth, IEEE 802.11 and Hyperlan makes possible the deployment of multi-hop ad hoc networks for commercial purposes. In these networks, because of the challenges introduced by wireless multi-hop transmission and limited resources, providing performance quality and reliability of data delivery comparable to wired networks is a major challenge.

Transmission control protocol (TCP) [1] is a connection-oriented unicast transport protocol that offers the following features: explicit and acknowledged connection initiation and termination; end-to-end reliability; in-order, and not duplicated data delivery; flow control; congestion avoidance; and out-of-band indication of urgent data. These multifold characteristics of TCP make it by far the most used transport protocol in Internet applications such as www (HTTP), mail (SMTP), file transfer (FTP) and remote access service (Telnet). In fact, TCP was originally designed to perform well in wired networks where the network congestion is the main reason for packet loss and it deals with this effectively by making corresponding transmission adjustment to its congestion window. At the present time, the Internet spans a very large base of heterogeneous networks such as wired and wireless, and ensuring that TCP provides efficient services for such a complex environment is a continuing mission of researchers.

In multi-hop wireless networks that employ IEEE 802.11 as the underlying network interface, TCP experiences severe performance degradation because it is not optimized to take advantage of the characteristics of wireless

* Correspondence: ammarhpc@gmail.com
[1]Department of Communication Technology and Network Faculty of Computer Science and Information Technology, Universiti Putra Malaysia 43400 UPM, Serdang, Selangor DE, Malaysia
Full list of author information is available at the end of the article

links. In wired networks, when a packet is detected to be lost, either by time-out or by multiple duplicated acknowledgments (ACKs), TCP slows down the sending rate to prevent network congestion. Unfortunately, wireless networks inherit several types of loss that are not related to congestion, which renders TCP not adapted for such an environment. The factors that may cause losses and affect TCP performance in multi-hop wireless networks are:

Frequent route failures caused by node mobility.

High bit error rates.

Medium access contention complicated by hidden/exposed terminal problems.

Interference and collision complicated by sharing the same path.

In fact, TCP is unable to distinguish between packet losses due to congestion from losses due to the specific features of multi-hop ad hoc networks. In theory, TCP should be independent of the underlying network technology. Specifically, TCP should not care whether it is running over wired or wireless connections. In practice, TCP suffers a significant degradation in performance over wireless networks because most TCP deployments have been carefully designed based on assumptions that are specific to wired networks and do not hold in a wireless environment. Ignoring the properties of wireless transmission can lead to poor performance of TCP. Therefore, numerous enhancements and optimizations have been proposed over the last few years to improve TCP performance over multi-hop wireless networks.

Overview of the article

The main aim of this review is twofold: the first is to present a comprehensive survey on research contributions that investigate the performance of TCP in multi-hop wireless networks; the second aim is to define existing research challenges, and to highlight promising new research directions. The scope of the survey is the core methods of improving TCP performance in multi-hop wireless networks, which encompass methods and protocols of TCP over multi-hop networks, including cross layer approaches. The research contributions in each field are systematically summarized and compared, allowing us to clearly define existing research challenges, and to highlight promising new research directions. The findings of this review should provide useful insights into current multi-hop networks literature and be a good source for anyone who is interested in "TCP in wireless" approaches or related fields. To this end, we first introduce the role of TCP in providing reliable end-to-end data transfer functions, and describe how TCP incorporates numerous control functions that are intended to make efficient use of the underlying network through a host-based congestion control function. Then, we present taxonomy of existing solutions, and describe their most representative features, benefits and design challenges. We also discuss open issues in this research area, with special attention to the ones most related to multi-hop wireless ad hoc networks.

In this article, we surveyed the solutions that guarantee the end-to-end data delivery that require no intermediaries to inform TCP about the state of the connection in wireless networks. The end-to-end semantic property of TCP provides the ultimate reliability at the TCP layer. This property would be violated if any intermediate node (not the TCP destination) generates ACKs on behalf of the destination. Any modifications that break the end-to-end semantic of TCP, such as splitting a TCP connection into wired and wireless portions so that the traffic control is done separately, cannot guarantee the arrival of a certain data segment at the destination even though the source has received the ACK of that segment. Thus, we only considered the solutions that preserve the end-to-end semantic of any established TCP connection so that the traffic control and maintenance of the connection are performed by the two end systems of the connection, based on the measured network conditions. Any solutions that require intermediaries or do not preserve the end-to-end semantic of a TCP connection are not discussed here. Interested readers can refer to [2,3] for early works to improve the performance of TCP in wired-cum-wireless environments. They can also refer to [4-8] for surveys of recent TCP enhancements in ad hoc wireless networks.

Another contribution of this article is to give the readers a new angle from which to view the existing state of the art by classifying the surveyed solutions based on the way they tackle the problems in multi-hop wireless networks. In [2], the authors explained the TCP's congestion control mechanism, the typical characteristics of heterogeneous wireless networks, and the problems that pose a threat to the traditional TCP implementation. Then, they presented a survey of the various schemes that try to solve these problems, classifying them according to their characteristics. In [3], the authors categorized the proposed mechanisms as the link-layer solutions, split solutions, TCP modifications, new transport protocols and wireless application protocol. In [4], the authors classified the surveyed proposals based on the layers where the enhancements have been implemented. In [5], the authors summarized the schemes that attempt to achieve better TCP performance with either of the two ideas: TCP should be capable of distinguishing non-congestion-related packet losses from congestion caused packet losses such that corresponding actions can be taken to deal with the losses; or non-congestion-related losses should be reduced such that TCP can work normally without any modifications. They choose to present the proposed schemes after separating them to two groups, the one-hop wireless networks and the multi-hop wireless networks. The aims of their survey are to explain each of the selected solution and show some of the future

directions. The authors in [6] used a detection-response framework to categorize different approaches and analysed the possible design options. They only selected the approaches that fit in their framework. In addition, they summarized the performance studies of these approaches. The purpose of their survey is not to compare them among each other, but to briefly show how well each approach performs in terms of improvement over standard TCP. In [7], the authors grouped the solutions based on whether a connection is split between wired and wireless portions as well as the way a wireless segment loss is handled. In [8], the focusing was on the cross-layer-based approaches that improve the performance of TCP in multi-hop mobile ad hoc networks. In this survey, no comparison between the solutions was presented. The survey briefly shows the problem identified and the solution proposed of each approach. Our survey article provides the readers a short tutorial of the surveyed representative solutions so that they can understand the basic mechanisms of these enhancements easily. Furthermore, it highlights the advantages and the weaknesses of the proposed solutions and discussed wither these solutions have alleviated the identified problem or they still need further improvements. In other words, our intended readers are the general audience who would like to quickly acquire the state of the art on TCP solutions in wireless networks.

The remainder of this article is organized as follows. Section 2 introduces a background about TCP, and TCP challenges in wireless networks. Section 3 categorizes and reviews core methods in multi-hop ad hoc networks that have been proposed to improving TCP performance. Section 4 shows a general comparison of the TCP approaches and their targeted features, discusses the strengths and limitations of these approaches, and identifies future research trends and challenges. Section 5 concludes the article.

2 Background on TCP challenges in multi-hop wireless networks

In this section, we begin with a brief overview of multi-hop ad hoc networks and some of its specific properties. Then, we give an overview of TCP and its functions. Finally, we present the problems that arise with TCP when applied over multi-hop ad hoc networks.

2.1 Overview of multi-hop wireless networks

With the recent proliferation of wireless communication and personal computing systems, wireless ad hoc networks are expected to play an important role in future civilian and military settings where wireless access to wired backbone is either ineffective or impossible. Ad hoc networks comprise mobile or static hosts that can communicate with each other using wireless links. It is also possible to have access to some hosts in a fixed infrastructure, which, in this case, is called hybrid network. In this environment, a route between two hosts may consist of one or more hops of ad hoc nodes. In this way, multi-hop network paths can be established between any pair of nodes without relying on a pre-existing network infrastructure, centralized control or dedicated network devices (i.e., router switches, servers).

Some of the most important issues in ad hoc networks are finding and maintaining routes, as host mobility can cause topology changes; efficient transport and reliable data delivery; and network security. These networks can be realized by different wireless communication technologies such as Bluetooth, IEEE 802.11, and Ultra-wide band (UWB). However, each one of these networks combined with the communication technologies pose various challenges in designing proper algorithms for them. As a result, several algorithms and considerable research efforts have been attempted to solve these problems. In this article, our focus is on TCP over multi-hop wireless networks that employ IEEE 802.11 as the underlying network interface.

2.2 Overview of TCP

A number of mechanisms is used by TCP to achieve high performance and avoid congestion collapse. These mechanisms maintain the sending rate of data entering the network at an appropriate level. Basically, the congestion window at the sender and the advertised window at the receiver regulate the maximum number of outstanding packets that can be transmitted. Modern implementations of TCP contain several intertwined algorithms such as slow-start, congestion avoidance, Fast Retransmit and Fast Recovery (RFC 5681) [9]. In addition, TCP sender employs retransmission time-out (RTO) mechanism, which is based on the estimated round-trip time (RTT) between the sender and receiver, and the variance in this round trip time. The behaviour of this timer is specified in RFC 2988 [10]. Enhancing TCP to reliably handle loss, minimize errors, manage congestion and go fast in very high-speed environments are ongoing areas of research and standards development. As a result, there are a number of TCP algorithms variations. As it is necessary to understand the performance of TCP when applying over ad hoc networks, this sub-section provides a brief overview of the main mechanisms of TCP, as well as discussing various acknowledgment generation methods. For a detailed description of TCP's algorithms, refer to reference [11].

Slow-Start

The principle behind this mechanism is to control TCP connection first initiated (or after a prolonged disconnection). It sets the congestion window (*cwnd*) size to "one" and by this, only one packet is allowed to be sent. Then, during this phase, *cwnd* increases exponentially over time for each arrival of a non-duplicate (or unique)

acknowledgment (ACK). This happens until either a packet loss event occurs or *cwnd* reaches the slow-start threshold (*ssthresh*). The benefit of this mechanism is to estimate the available bandwidth by progressively probing for more bandwidth. If *cwnd* reached the *ssthresh*, TCP enters the linear growth phase or called congestion avoidance phase. During the slow start phase, when a loss event occurs, TCP assumes this is due to network congestion and takes steps to reduce the size of *cwnd*, which, in turn, slows down the sending rate. After the first packet loss, *ssthresh*, which is used to determine the start of the congestion avoidance phase, is set to half of the current *cwnd*. Then, *cwnd* will be reset to one and will grow according to the aforementioned procedure until *ssthresh* is reached. Although the strategy is referred to as "slow-start", its congestion window growth is quite aggressive, more aggressive than the congestion avoidance phase. Figure 1 gives an overview of slow start behaviour.

Congestion Avoidance

This phase, as its name implies, tries to avoid network congestion by restricting the growth of *cwnd*. In this way, TCP forces a linear increase of *cwnd* after it reaches the *ssthresh*. This linear increase is achieved by incrementing the *cwnd* additively by one segment for each RTT and an ACK is received. This happens until a loss event occurs. Figure 1 shows an example of how the congestion window is controlled by the slow-start and the congestion avoidance algorithms for an *ssthresh* of 16.

Multiplicative decrease

It is the algorithm that controls the *ssthresh* value, which is, in turn, responsible for changing the slow start phase to the congestion avoidance phase. With a

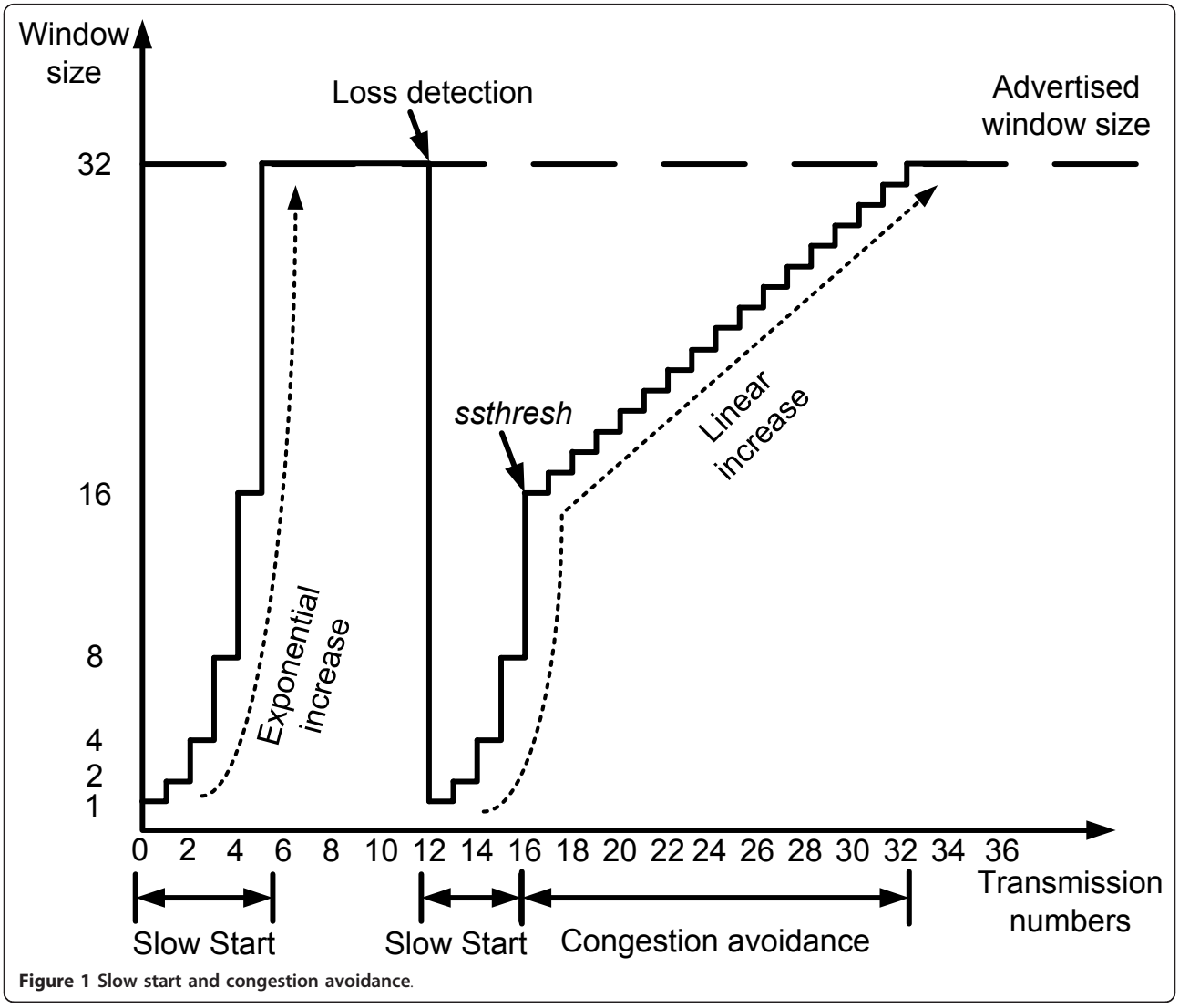

Figure 1 Slow start and congestion avoidance.

multiplicative decrease, TCP sets *ssthresh* to half of the current *cwnd* each time a loss event occurs. To this point, the *cwnd* itself is set to one to force a slow start. Thus, in case of consecutive loss, multiplicative decrease reduces the sending rate severely.

Fast Retransmit and Fast Recovery

Identifying the packet loss in TCP is indicated by two means. One is the occurrence of a time-out event, which corresponds to a failure of receiving an ACK within a predefined time interval (RTO). Nevertheless, when the RTO value is large, waiting for a time-out event to trigger retransmission is considered inefficient. Due to this, "Fast Retransmit" was proposed to provide a timely detection of a lost packet. It triggers when the sender receives three duplicate ACKs from the receiver. Upon receiving these duplicate ACKs, TCP retransmits the missing packet. Fast Recovery is executed after Fast Retransmit: It first sets *ssthresh* to half of the current *cwnd* and then reduces *cwnd* to *ssthresh*. When a non-duplicate acknowledgment is received, TCP exits the Fast Recovery, sets *cwnd* equal to *ssthresh*, and enters the congestion avoidance phase.

Generating Acknowledgements

The mechanisms described above are a sender-based mechanism. They must deal with a TCP receiver mechanism, which is responsible for returning ACKs to the TCP sender. Currently, there are several forms of acknowledgment, which can be used alone or together in TCP.

Positive Acknowledgement The receiver explicitly notifies the sender which packets were received correctly. In this way, the receiver will know which packets were not received and need to be retransmitted. This method is used by TCP (RFC 793) [1] to verify receipt of the transmitted data. Most of the TCP variants are based on this mechanism where a positive ACK should be generated for each packet received.

Negative Acknowledgement (NACK) In this form, the receiver explicitly notifies the sender which packets were received incorrectly and thus may need to be retransmitted (RFC 4077) [12].

Selective Acknowledgement (SACK) In this mechanism, the receiver should inform the sender about all segments that have arrived successfully. In this way, the sender will only retransmit the segments that have actually been lost. Positive selective acknowledgment is an option in TCP (RFC 2018) [13].

Cumulative Acknowledgement Each ACK is a confirmation that all packets up to the current number have been received correctly.

Delayed Acknowledgement Instead of generating an ACK as soon as a packet is received, this mechanism gives the TCP receiver the ability of delaying the ACK for a while. Specifically, as introduced in RFC 1122 [14], an ACK should be generated for at least every second in-order data packet received. The receiver contains a timer bound variable interval (default 500 ms) that gives the number of seconds to wait for the second in-order packet. Thus, the ACK must be generated within 500 ms of the arrival of the first unacknowledged packet. The time-out of this timer causes immediate ACK generation of the first packet. In addition, out-of-order packets cause immediate ACK generation.

Duplicate Acknowledgement In order to trigger the fast retransmit algorithm, the receiver should send an immediate duplicate ACK when it receives a data packet above a gap in the sequence space.

2.3 TCP challenges in multi-hop wireless networks

In this section, a thorough understanding of the major problems that arise with TCP, when applied over multi-hop wireless networks, is discussed.

2.3.1 Routing failures

In stable wired networks, route failures occur very rarely. However, in multi-hop wireless networks they are the norm rather than the exception. The node mobility is the main source of frequent topology changes and route failures in mobile ad hoc networks. Moreover, the link failures due to the contention on the wireless channel may lead to route failures in both static and mobile ad hoc networks. When a route failure occurs, packets that are buffered at intermediate nodes along the route will be dropped. This large amount of packet drops may cause a series of time-outs at the TCP sender. As a result, the RTO value will be doubled for each subsequent time-out. Furthermore, TCP will mistakenly interpret the loss as an indication of network congestion and trigger the congestion control mechanisms to reduce the size of *cwnd* and *ssthresh*. In addition, TCP does not have any indication of the route re-establishment duration, because the route re-establishment event after route failures depends on the underlying routing protocol. These actions have three adverse effects: (1) the small *cwnd* and *ssthresh* values reduce the initial sending rate after the route is restored. Therefore, it takes a long time for the sending rate to catch-up to a high value after a new route is found. (2) The large RTO value reduces the responsiveness of TCP; even if the route is restored, TCP will take a long time to converge to the right level of operation. (3) The large idle time of TCP caused by route re-establishment will degrade the throughput.

2.3.2 Random wireless losses

The medium transmission errors in wired networks so-called bit error rate (BER) are negligible in contrast with multi-hop wireless networks. Channel errors of wireless usually occur randomly and in bursts due to channels fading and interference [15], which cause it to be error-prone in nature. High BER introduces two challenges to

the reliable TCP protocol. (1) Packets loss due to corruption violate the assumption of TCP's congestion control mechanism in that congestion is the main reason for most losses. In this case, whenever a packet loss is detected, TCP will mistakenly take this as an indication of network congestion and reduce the sending rate unnecessarily. In other words, even though the network is not congested, TCP reduces the sending rate in the face of wireless errors. This unnecessary reduction of sending rate deteriorates TCP throughput when packet losses are mainly due to corruption. (2) In the presence of severe wireless channel contention, where the BER is high and the link-layer local recovery mechanism [16] is unable to recover the lost packets, TCP will face a large amount of consecutive packet losses. The current TCP's "error detection and recovery" mechanism is inefficient to recover from these losses. Either only one lost packet is recovered per RTT, or a time-out is needed to recover from substantial losses.

2.3.3 Hidden and exposed terminals

Due to the sharing of the wireless bandwidth among ad hoc stations, medium access control (MAC) may rely on physical carrier sensing multiple access mechanism with collision avoidance (CSMA/CA) to determine the idle channel, such as in the IEEE 802.11 distributed coordination function (DCF). However, all CSMA/CA-based MAC protocols do not completely solve the hidden/exposed station problem [17]. Figure 2 illustrates an example of the hidden terminal problem. Suppose that nodes A and C want to transmit to node B. By only sensing the medium, node A will not be able to detect transmissions by any node in the dashed area, such as C, because it is outside the transmission range of C. Node C is, therefore, "hidden" to node A. Thus, transmission of A and C will lead to collisions at node B. Unfortunately, this will affect TCP and lead to poor performance.

To alleviate the hidden station problem, virtual carrier sensing has been introduced to wireless MAC layers [18].

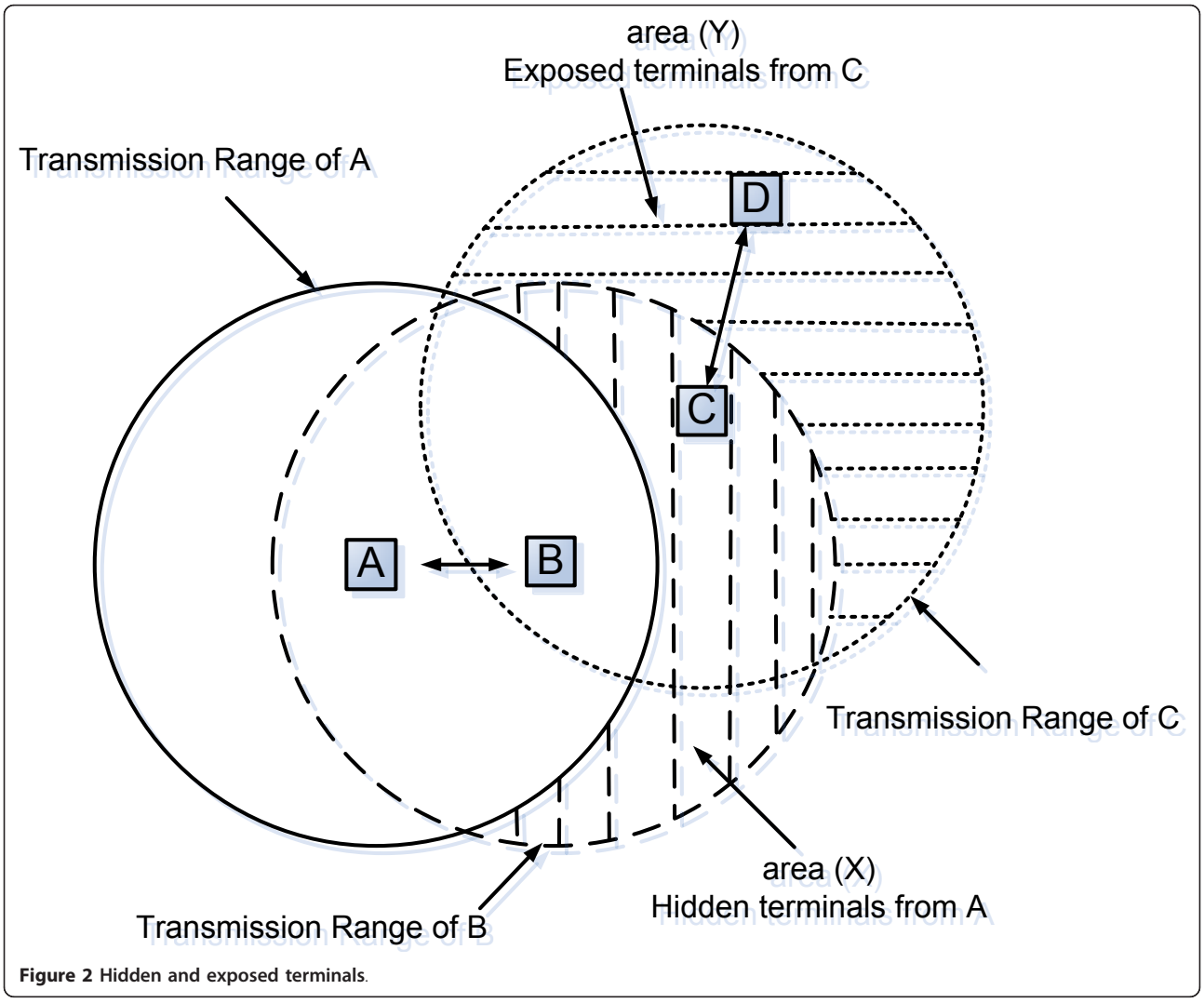

Figure 2 Hidden and exposed terminals.

It is based on a two-way handshaking that precedes data transmission. One such virtual sensing mechanism is the 802.11 Request To Send/Clear To Send (RTS/CTS) exchange mechanism [19]. In this mechanism, the source station transmits a short control frame, called RTS, to the destination station. Upon receiving the RTS frame, the destination station replies by a CTS frame, indicating that it is ready to receive the data frame. Both RTS and CTS frames contain the total time of the data transmission. All stations receiving either RTS or CTS will keep silent during the data transmission period (e.g., station C in Figure 2).

However, as pointed out in [20,21], the hidden station problem may persist in IEEE 802.11 ad hoc networks even with the use of the RTS/CTS handshake. This is because the power needed for interrupting a packet reception is much lower than that of delivering a packet successfully. In other words, the node's transmission range is smaller than the sensing node range. Moreover, the RTS/CTS mechanism introduces a new problem termed the exposed terminal problem, where some nodes that heard the RTS/CTS exchange refrain from transmission even though they would not have interfered with any ongoing transmission. In Figure 2, we show a typical scenario where the exposed terminal problem occurs. Let us assume that node A wants to transmit to node B. Node A sends an RTS and waits for B to send a CTS. Suppose a node D is transmitting data to node C, thus, D has transmitted an RTS to C just before A sends the RTS to B and C has transmitted a CTS. This CTS is heard by B which prevents B from sending the CTS to A. Therefore, any transmission from a node within the area (Y) to a node within (X) will prevent A from transmitting data to B, although simultaneous transmissions from area Y to × would not have interfered with transmission from A to B. Thus, the exposed station problem may result in a reduction of channel utilization. Unfortunately, this will affect TCP performance and lead to increased packet transmission delays. Moreover, this allows an aggressive sender to capture the channel, which reduces the chance of transmissions of the other senders in the vicinity.

2.3.4 Intraflow and interflow contention: packets compete for airtime

In shared multi-hop wireless networks, nodes cooperate to forward each other's packets through the networks. Due to the contention for the shared channel, the throughput of each single node is limited not only by the raw channel capacity, but also by the transmissions in its neighbourhood. To discuss the impact of medium contention on the performance of TCP traffic, the channel contentions are characterized as intra-flow contention and inter-flow contention, which result from the interaction between the TCP traffic and the MAC layer contentions.

Inter-flow contention refers to the contention experienced by a node due to transmissions by nearby flows. Intra-flow contention (or self-contention [22]), however, refers to the contention for the shared channel experienced by a node due to the transmissions of the same flow (due to the forward data transmissions and the reverse ACKs transmissions). Thus, each multi-hop flow encounters contentions not only from other flows that pass through the neighbourhood, i.e., the inter-flow contention, but also from the transmissions of itself because the transmission at each hop has to contend the channel with upstream and downstream nodes, i.e., the intra-flow contention.

The following discussion demonstrates the effects of both types of contention. In Figure 3, we show a typical scenario where the intra-flow contention problem occurs. Consider a chain topology with seven nodes and the sender is placed at the first node (node 0). The transmission of node 0 in a 7-node chain experiences interference from three subsequent nodes, while the transmission of node 2 is interfered by five other nodes. This implies that node 0, i.e., the source, could actually inject more packets into the chain than the subsequent nodes can forward. These packets are eventually dropped at the two subsequent nodes. Besides the contentions inside a multi-hop flow, the contentions between flows could also seriously decrease the network throughput. If two or more flows pass through the same region, the forwarding nodes of each flow encounter contentions not only from its own flow but also from other flows. Thus, the previous hops of these flows could actually inject more packets into the region than the nodes in the region can forward. These packets are eventually dropped by the congested nodes.

Both types of contention could result in severe collisions and congestion, and significantly degrade the performance of TCP in multi-hop ad hoc networks [23]. Moreover, contention allows an aggressive sender to capture the channel, which reduces the chance of transmissions of the other senders in the vicinity. This not only greatly decreases the end-to-end throughput but also increases the end-to-end delay due to the long queuing delay. Furthermore, heavy contention is usually caused by using a large TCP congestion window. Previous studies have shown that the maximum congestion window size should be kept small and should be maintained at a level proportional to some fraction of the length (in terms of the hop count on the path) of the connection [20,24] in order to alleviate the effect of contention. Unfortunately, as indicated in reference [20], conventional TCP does not operate its congestion window at an appropriate level and, thus, its performance is affected by the contention severely.

2.3.5 TCP instability and delay spike

The RTT estimation of TCP is adequate in a stable network in which the RTT fluctuations are small. Thus, through several RTTs, the rate of TCP can be increased

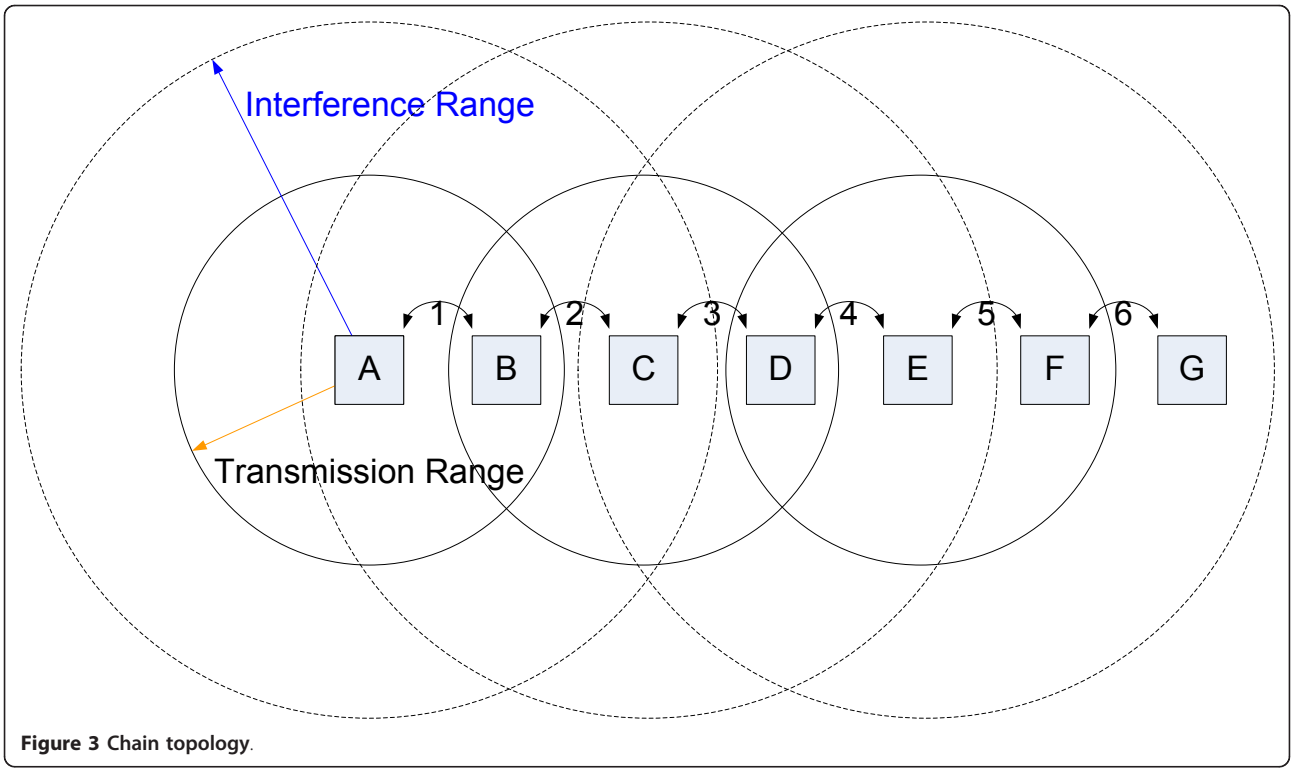

Figure 3 Chain topology.

to a high value for better performance. However, the multi-hop wireless networks may often experience delay spikes in the order of up to a few seconds [25]. Delay spike is defined as a sudden and sharp delay increase exceeding the typical RTT by several times of the typical TCP connection. Delay spikes will cause TCP to invoke spurious retransmissions.

There are many factors that may cause a sudden delay spike to be frequent in wireless multi-hop networks, such as: (1) hop-by-hop link-layer errors and contention along the path that leads to a longer waiting time and many link-level retransmission attempts before the packets in the queue are transmitted, (2) route changes or intermittent disconnections due to mobility that leads to a higher delay experienced by the transmitted packets,(3) wireless bandwidth fluctuation and (4) blocking by high-priority traffic.

In TCP, a retransmission timer mechanism is adopted to ensure reliable delivery of data. The retransmission time-out value is dynamically determined by estimating the RTT samples of the connection. Under the above circumstances, when a delay spike occurs the sender may not receive an acknowledgment within the time-out period and, thus, the sender will regard all the transmitted but not acknowledged packets as lost. Consequently, the sender will unnecessarily reduce the sending rate and aggressively retransmit those packets that are deemed lost but are merely delayed. Meanwhile, TCP will exponentially increase the retransmission time-out value, which leads to a long waiting time period if congestion loss takes place and, thus, wastes the limited available bandwidth.

3 Approaches to improve TCP performance in multi-hop wireless networks

In this section, we present some of the major approaches that have been made in the literature to improve the performance of TCP in multi-hop wireless networks. We regroup these approaches to four sets according to the strategy used in order to improve TCP performance. However, some approaches can be matched to several types of strategies but are only classified to their main strategy. The approaches that belong to the same set are classified into two types: cross-layer approaches and layered approaches. The cross-layer proposals rely on interactions between two layers of the open system interconnection (OSI) architecture. Specifically, the cross-layer interaction, considered in this article, is between the TCP layer and either the network layer or the data link layer. These approaches were motivated by the fact that "providing lower layer information to upper layer should help the upper layer to perform better". The layered approaches rely on adapting OSI layers independently of other layers. Thus, depending on which layer is involved, layered proposals can be further classified into three types: TCP layer, network layer and link layer approaches. For each approach presented here, the representative schemes are

presented and their mechanisms are described, along with discussion concerning their strengths and weaknesses in improving TCP performance in multi-hop wireless networks. In addition, a comparison of the main characteristics of various TCP enhancements is provided for each category.

3.1 Determining the routing failures

In typical wired networks where the route for the connection is relatively invariable, TCP performs well and network congestion is the key factor affecting its performance. However, in multi-hop wireless networks, route failures are the norm rather than the exception and occur frequently due to mobility and high BER. When a route failure occurs, packets that are buffered at the intermediate nodes along the route will be dropped. Moreover, the routing protocol would take time to discover a route-route re-computation process. The amount of time required for retransmitting the dropped packets and for discovering a new route has a negative impact on TCP performance. In addition, when TCP detects the bursty losses, TCP sender will encounter a time-out or even series of time-outs. In this case, TCP will dramatically reduce the size of *cwnd* and *ssthresh* because it does not have the ability to distinguish between losses due to route failure and those due to congestion. Therefore, it takes a long time for the sending rate to return to a high value after a new route is restored.

Due to the above factors, TCP needs to have some effective means to determine these events accurately so as to allow it to react appropriately. The approaches that address the problem of TCP caused by route failure can fall into three categories: TCP layer approaches, network layer approaches, and TCP and network cross-layer approaches.

3.1.1 TCP layer approaches

Fixed RTO Dyer et al. introduced fixed RTO [26], which is a simple heuristic technique that does not rely on feedback from the underlying layers. Fixed RTO is designed to distinguish between route loss and network congestion and thereby tries to improve the performance of TCP. In the event of time-out, the conventional TCP retransmits the unacknowledged packet and doubles the RTO interval. For each retransmission of the packet, the RTO is doubled until an ACK for the retransmitted packet has been received. This exponential growth of the RTO enables TCP to handle network congestion gracefully. However, in multi-hop wireless networks, the losses may occur frequently and temporarily due to route losses and the network congestion is rare. Based on this and by taking advantage of the capability of routing algorithms that are designed to repair broken routes quickly, the authors show that it is intuitive to let TCP retransmit the unacknowledged packet at periodic intervals rather than having to wait increasingly long periods of time between retransmissions. Therefore, fixed RTO keeps the RTO fixed after the first retransmission. When time-outs occur consecutively, i.e., the missing ACK is not received before the second RTO expires, this is taken to be evidence of a route loss. The unacknowledged packet is retransmitted again but the RTO is not doubled a second time. The RTO remains fixed until the route is re-established and the retransmitted packet is acknowledged.

To evaluate the performance of TCP with fixed RTO, three routing protocols, two on-demand (Ad-Hoc on Demand Distance Vector (AODV) and Dynamic Source Routing (DSR)) and one adaptive proactive routing protocol (Adaptive Distance Vector (ADV)), have been considered. The results indicate that the proactive ADV algorithm performs well under a variety of conditions and that the fixed RTO technique improves the performances of TCP over the two on-demand algorithms significantly. Nevertheless, the assumption that two consecutive time-outs are the exclusive results of route failures needs more analysis, especially in the presence of congestion.

TCP DOOR TCP detection of out-of-order and response (DOOR) [27] is an end-to-end approach which does not require any feedback from the network or from the lower layers. This approach was designed by Wang and Zhang to improve TCP performance by detecting and responding to out-of-order packet delivery events, which are the results of frequent route changes. In TCP-DOOR, the out-of-order (OOO) events are interpreted as an indication of route failure. Thereby, the sender can distinguish route changes from network congestion. The detection of OOO events can be accomplished either by the sender or by the receiver. While the receiver can notify the sender about detected out-of-order data packets, the sender itself may notice ACKs arriving out-of-order. Once the TCP sender recognizes the OOO event, two response actions are suggested. In the first action, the sender may temporarily disable the congestion control mechanism of TCP by keeping its state variables constant. In the other action, if the congestion control mechanism was invoked during the past time period, the TCP sender should recover immediately to the state before the invocation of the congestion control.

In TCP-DOOR, the authors recommend their approach primarily for an environment having both ad hoc and fixed infrastructural networks where the adaptation of a feedback-based approach is particularly hard. In general, TCP DOOR shows significant improvements over standard TCP. Nevertheless, more analysis is required to answer the question of what if OOO is not caused by route changes. In fact, route changes are not the only reason for out-of-order packets delivery. For example, in multi-path routing protocol such as Temporally-ordered

routing algorithm (TORA) [28], OOO may occur if packets from different paths are out-of-order.

3.1.2 Network layer approaches

Backup Path Routing In this approach, Lim et al. [29] have investigated the performance of TCP over a multipath routing protocol. It is shown that multipath routing can improve the path availability of TCP connections. The authors found that the original multipath routing may degrade TCP performance due to frequent out-of-order packet delivery and inaccuracy in average RTT measurement. Therefore, they introduce a new variation of multipath routing strategy, called backup path routing. Under the backup path routing strategy only one path at any time is used. However, the backup path routing strategy maintains several paths from source to destination. The other alternative paths are used as a backup when the current path is broken. Two methods have been considered as criteria for selecting the paths. In the first method, the shortest-hop path is selected as the primary path and the shortest-delay path is the alternative one. For the second method, the shortest-delay path is the primary and the maximally disjoint path is selected to be the alternative. The alternative maximally disjoint path is the path that has the fewest overlapped intermediate nodes with the primary path.

The simulation results show that TCP is able to gain improvements under the backup path routing scheme. As a comparison between the two methods suggested in this approach, the authors found that the first method outperforms the second one. They justified that when the second method is used, routes tend to be longer in the number of hops. Nevertheless, the method where the shortest-path is selected as primary and maximal disjoint as alternative was not considered. Moreover, the authors show that TCP's performance degrades when the multipath routing protocol SMR [30] is used.

Routing exploiting multiple interfaces In [31], Yoon and Vaidya proposed a network layer scheme that utilizes multiple heterogeneous wireless interfaces in multi-hop wireless networks. The idea is to use two heterogeneous wireless interfaces in each node: a primary 802.11a interface and a secondary 802.11b (or 802.11) interface. The primary path over 802.11a interface is maintained by a reactive routing protocol such as DSR, while the secondary path over 802.11b interface is maintained by a proactive routing protocol such as Destination-Sequenced Distance-Vector Routing (DSDV). In normal conditions, TCP data packets will use the primary path, which has the higher rate, and the control packets (ACK) will take the secondary path. In the event of link failures, TCP will use the secondary path to recover its packets and preserve its window size. It keeps communicating using the low-rate interface, until the high-rate path is recovered. The 802.11b (or 802.11) interface is used at the secondary path because they exhibit different properties than the 802.11a interface (i.e., slower rate but larger transmission range).

The simulation results show that using a secondary, lower rate but longer range interface can increase TCP throughput. However, the proposed scheme requires that each node be equipped with two radio interfaces, which raises the question of whether the additional cost pays off the performance gain. Moreover, dynamic route changes induced by node mobility causes reordering of TCP packets at the receiver side, which, in turn, incurs duplicate ACKs at the sender side. With multiple interfaces routing, the chance of TCP packet reordering is higher and may affect the potential gain of using multiple interfaces. A more intelligent delivery mechanism to avoid TCP packet reordering is required.

3.1.3 TCP and network cross-layer approaches

Explicit link failure notification technique (ELFN) ELFN [32] is a simple approach designed by Holland and Vaidya. This approach is based on a real interaction between TCP and routing protocols. This interaction aims to provide link failure information to the TCP sender to allow TCP to distinguish packet losses that are caused by link failures from those that are caused by congestion. When a link failure occurs, an ELFN message, which is piggybacked on the route failure message, will be sent by the routing protocol to the TCP sender. The ELFN message is like a "host unreachable" Internet control message protocol (ICMP) message, which contains the sender receiver addresses and ports, as well as TCP packet's sequence number. The sender, upon receiving the ELFN message, responds by disabling its retransmission timers and enters a "standby" mode. During the standby period, the TCP sender uses a periodic probe message to determine whether the route has been restored. If the acknowledgment of the probe packet is received (implying that the route is re-established), the TCP sender resumes its retransmission timers and leaves the standby mode to continue the normal operations. In this way, TCP can avoid the slow-start phase and continues with a high rate.

Explicit link failure notification technique (ELFN) is a simple efficient technique that provides significant enhancements over standard TCP. However, ELFN requires intermediate nodes to notify TCP on the presence of route failures and this makes its deployment and implementation complicated.

3.1.4 Comparison

Six approaches have been presented. These approaches address the problem of TCP inability to distinguish between losses due to route failures and network congestion. The discussion proceeds with TCP layered solutions, like fixed RTO, and TCP-DOOR, network layer solutions such as Backup Path Routing, and Routing Exploiting Multiple Interfaces, and then a TCP and network cross-layer solution, like ELFN. In TCP layer approaches, fixed

RTO and TCP DOOR employ end-to-end TCP semantic techniques to distinguish between packet losses induced by route failures and network congestion. In fixed RTO, this is done by considering two consecutive time-outs as a sign of route failures. Meanwhile, in TCP-DOOR, receiving out-of-order packets is interpreted as an indication of route failures. The main advantage of fixed RTO and TCP DOOR is that they do not require any notification from routing layer neither do they require other nodes cooperation to detect route failures. Comparing these two proposals, TCP DOOR performs better than fixed RTO, but at the cost of more modifications.

In the network layer solutions, Backup Path Routing and Routing Exploiting Multiple Interfaces maintain two routes, one primary path for normal data transmission and another backup path for recovering the data in case of route failures. In Backup Path Routing, this is done by using multipath routing protocol. Meanwhile, Routing Exploiting Multiple Interfaces uses different routing protocol running over different radio interfaces. By comparing these two approaches, Routing Exploiting Multiple Interfaces performs better than Backup Path Routing, but at the cost of employment where each node must be equipped with two radio interfaces. The TCP and network cross-layer proposal, ELFN, is based on an explicit notification from the network layer to detect the route failure. For detecting re-establishments, ELFN uses a probing mechanism and the implementation of this mechanism is not easy. What is the optimal value of the probing interval? And what is the implication of this mechanism in the case of high load? Especially, it is seen that in the case of high load, ELFN performs worse than the standard TCP. Table 1 illustrates a summary of the main characteristics of the discussed TCP enhancements.

3.2 Estimating bandwidth and channel status

As discussed in Section 2, the traditional loss-based congestion control mechanism of TCP cannot accurately adjust the sending rate when it is used in multi-hop wireless networks. Packet loss is not always a sign of congestion; it could be due to mobility or due to wireless errors. It is well known that one of the critical sources of TCP poor performance in multi-hop wireless networks lies on the lake coordinate between TCP and MAC layers. If there exists a big difference between the transmission rates of MAC and TCP, it may cause network congestion and retransmissions. To this end, several TCP schemes have been proposed recently to address the problem by better estimating the available bandwidth and the channel status. The representative schemes that take advantage of controlling the rate to enhance TCP performance include TCP-Vegas [33], TCP-Westwood [34], TCP-CL [35], Channel Efficiency-Based Transmission Rate Control [36], and TCPCC [37].

3.2.1 TCP layer approaches

TCP-Vegas TCP-Vegas [33] is one of the TCP variants that uses a rate-based congestion control mechanism. It was developed at the University of Arizona by Brakmo and Peterson. The main idea is to adjust the sending rate carefully by comparing with the estimated rate. It emphasizes packet delay, rather than packet loss, as a sign to help determine the rate at which to send packets. In addition, it also modifies the congestion detection and avoidance algorithms to improve the overall throughput of TCP. Even though TCP-Vegas was not intentionally designed for wireless, the TCP performance can be improved because its inherent rate-based congestion control algorithm could pro-actively avoid possible congestion and packet losses by ensuring that the number of outstanding segments in the network is small.

The most significant difference between TCP-Vegas and the conventional TCP variants is the use of a rate-based technique to control the congestion window size. Unlike other flavours like Reno, NewReno, etc., which detect congestion only after it has actually happened via packet drops, TCP Vegas detects congestion at an incipient stage based on increasing RTT values of the packets in the connection. For every RTT, TCP-Vegas compares the expected throughput to the actual throughput measured. The expected throughput is calculated as the current window size divided by the minimum observed RTT. However, the actual throughput is measured as the number of bytes transmitted between the time a distinguished segment is transmitted and acknowledged, divided by the time it takes to get the acknowledgment back. If the difference between the two values is smaller than α, TCP-Vegas increases *cwnd* linearly for the next RTT, assuming that throughput is less than the available bandwidth; and if the difference is greater than β, TCP-Vegas decreases the congestion window linearly for the next RTT to avoid overrun the bandwidth. If the difference is between α and β, TCP-Vegas keeps *cwnd* unchanged. In addition to the rate-based congestion control modification, TCP-Vegas modified the slow-start mechanism by allowing *cwnd* to grow exponentially only once in every other RTT. This is to allow TCP-Vegas to compare the expected and the actual throughput. If the difference is greater than the γ threshold, TCP-Vegas changes from slow-start mode to the linear increase/linear decrease mode as described above.

However, TCP-Vegas also inherits some of the weaknesses of conventional TCP. It cannot handle the effects of route failure and wireless channel errors. Moreover, as reported in [38], TCP-Vegas suffers from several other problems. First, since TCP-Vegas uses baseRTT for calculating the expected throughput, in wireless network, the baseRTT may not reflect the actual minimum measured round-trip time of the connection due to route change in multi-hop wireless networks. Therefore, there

Table 1 Comparison of the main characteristic of various TCP Enhancements

	"Fixed RTO"	"TCP-DOOR"	"BACKUP PATH"	"ROUTING EXPLOITING MULTIPLE INTERFACES"	"ELFN"
	[26]	[27]	[29]	[31]	[32]
DEALING WITH ROUTING FAILURES	Yes: the expiration of second RTO is taken to be evidence of a route loss.	Yes: out of order event is interpreted as an indication of route failure.	Yes: it uses multiple paths. if one path fail the other will backup the transmission	Yes: different routing protocols with different radio interfaces are used to backup that transmission	Yes: notify TCP about the route failure through ELFN message
DEALING WITH WIRELESS ERRORS	No.	No.	No.	No.	No.
DEALING WITH CONTENTION	No.	No.	No.	No.	No.
DEALING WITH RETRANSMISSION	Yes: it uses fixed RTO.	No.	No.	No.	Yes: upon receiving ELFN message, disable the retransmission timers and enters a standby mode.
DEALING WITH TCP CONGESTION CONTROL MECHANISM	No.	Yes: it temporarily disables the congestion control by keeping its state constant in case of out of order events.	No.	No.	No.
DEALING WITH HIDDEN EXPOSED PROBLEM	No.	No.	No.	No.	No.
DEALING WITH TCP RATE	No.	No.	No.	No.	No.
DEALING WITH TCP ACK	No.	Yes: ACK is used to notify the sender about out of order packet delivery.	No.	No.	No.

are inaccuracies in the calculated expected throughput after a route change event. Another problem is the unfairness of TCP-Vegas, when it inter-operates with other versions like Reno. In this case, the performance of Vegas degrades because Vegas reduces its sending rate before Reno as it detects congestion early and, hence, gives greater bandwidth to co-existing TCP Reno flows. Therefore, the fairness provided by the linear increase/decrease mechanism for congestion control in Vegas is an important issue for research.

TCP-Westwood is another TCP variant based on bandwidth estimation, which was introduced by Casetti et al. [34]. TCP-Westwood is a sender-side modification that improves upon the performance of TCP Reno in wired as well as in wireless networks. The key idea is to continuously estimate the bandwidth used by the connection via measuring the rate of returning ACKs. When a loss occurs, either by receiving three duplicated ACKs or after a time-out, TCP-Westwood uses the estimated bandwidth to capture the congestion state much faster and then computes the congestion window and slow start threshold. The authors call this mechanism faster recovery. Moreover, when there is a prolonged absence of ACKs, the estimated bandwidth will exponentially decrease. The rationale of this strategy is simple: upon a packet loss event, in contrast with conventional TCP, which blindly reduces the *cwnd* and *ssthresh*, TCP Westwood attempts to compute a proper *cwnd* and *ssthresh* values, which are consistent with the effective bandwidth used at the time congestion is experienced. In fact, the conventional TCP schemes are not effective over wireless links where sporadic losses due to radio channel problems are often misinterpreted as a symptom of congestion and, thus, lead to an unnecessary window reduction. Nevertheless, TCP-Westwood is particularly robust to wireless because it tries to maintain the sending rate at the level just before the occurrence of a packet loss to avoid the unnecessary window reduction.

Experimental studies of TCP-Westwood reveal improvements in throughput performance as well as in fairness. In addition, friendliness with TCP Reno was observed in a set of experiments showing that TCP Reno connections are not starved by TCP-Westwood connections. Moreover, TCP-Westwood fully complies with the end-to-end TCP design principle and the improvement is most significant in wireless networks with lossy links. However, in contrast with conventional TCP, which cannot distinguish between random errors and congestion loss, TCP-Westwood performance is also not sensitive to random errors. When a random error occurs, caused by route failure or by channel

error, the estimated bandwidth will quickly reduce as described previously and this leads to poor performance after recovering the path. Similar to TCP-Vegas, TCP-Westwood uses the observed smallest roundtrip time (RTTmin) in estimating the bandwidth. This may lead to problems, since any route change will invalidate the RTTmin and, thus, lead to incorrect bandwidth estimates. Thus, the tendency of the researches is to make TCP able to react to different types of error.

3.2.2 Cross-layer approaches

TCP-CL Cheng and Lin proposed TCP-CL [35], which is a collaborative approach based on a cross-layer design for enhancing the end-to-end performance of TCP in wireless networks. The authors show that by making slight modifications to the legacy IEEE 802.11 MAC and TCP protocols, TCP-CL provides a significant improvement in the performance of TCP in multi-hop wireless environments. The standard IEEE 802.11 MAC layer provides a reliable operation over the communication channel by defining a retry limit parameter (RETL). Using this parameter whenever a node fails to transmit a frame, it retransmits that frame and then increases the value of RETL by one. If the value of RETL exceeds a specified threshold (i.e., 7 for basic access mechanisms and 4 for virtual carrier-sensing mechanisms), the frame will be discarded, link failure will be reported to the link layer, and the RETL will be set to zero. The RETL is also reset to zero once the frame has been successfully transmitted. Nevertheless, if the link experiences a high degree of contention, the MAC layer may mistakenly infer a link failure. Consequently, this misinterpretation of link failures may severely affect the performance of TCP.

Thus, to reduce the effects of link-layer contention on the performance of TCP, this approach extends the IEEE 802.11 DCF scheme by introducing a new variable, designated as the retransmission limit (RETF), to record the number of retransmission attempts in the event of continuous transmission failures. If the value RETF does not exceed the retransmission threshold as well as receipt of TCP ACK within the same flow, it forwards the TCP negative acknowledgment (NAK) piggybacks using the reverse TCP ACK along the end-to-end path and increases the value of RETF. If the value of RETF exceeds the retransmission threshold, it discards the transmitted packet and then resets the contention and RETL, respectively. The NAK option is triggered only when a link-layer frame is dropped as a result of transmission errors and is limited by the retransmission threshold value. If a TCP data frame is discarded after several retransmission attempts (limited by the retry threshold), the MAC-layer protocol triggers the TCP NAK option in the TCP header associated with the sequence number of the dropped packet and then piggybacks the option using a reverse TCP ACK to notify the TCP sender to retransmit the missing packet. Note that the NAK notification is sent in piggyback mode with the return TCP ACK segment in order to avoid increasing contention in the link layer. This cross-layer support from the link layer protocol ensures that the transport layer TCP protocol is aware of the transmission error in the link layer and can then react to this error in accordance with the wireless corruption information conveyed by the received NAK. TCP is therefore capable of differentiating between corruption and congestion losses, thus, allowing it to react appropriately in each case.

This approach has proposed a cross-layer solution designated as TCP-CL to improve TCP performance in multi-hop wireless networks. Overall, the results show that the proposed scheme has a number of key advantages compared to conventional TCP, including a more efficient treatment of frequent transmission losses, a faster reaction to corruption losses, and the ability to distinguish between congestion errors and transmission errors and to take appropriate remedial action. Importantly, the proposed mechanism does not require any node to cache any unacknowledged packets for every TCP connection passing through it. The performance of TCP-CL is compared with that of standard TCP-Reno and TCP-Westwood schemes in terms of throughput. The simulation results reveal that TCP-CL achieves a significant improvement in the TCP transmission performance over multi-hop wireless networks. However, this scheme keeps a single path to each destination without utilizing available paths effectively. Moreover, more analysis is required to study the effect of mobility on this scheme.

Channel efficiency-based transmission rate control is a cross-layer approach conducted by Zhang et al. [36]. In this approach, the authors show that, in multi-hop wireless networks, if the transmission rate of TCP does not match that of the medium access control (MAC) protocol, it causes network congestion and network performance degradation. Therefore, they propose a new approach to control the transmission rate of TCP by utilizing the MAC information through a cross-layer. The main contribution of this approach is to feedback the real channel efficiency of MAC protocol to TCP so as to adaptively control the transmission rate. This will help to reduce the big difference between the transmission rate of TCP and MAC and, consequently, reduce the network congestion and retransmission. To this end, two important measures, real MAC channel efficiency and virtual TCP channel efficiency, have been introduced. In the criterion of the real MAC channel efficiency, TCP adjusts the transmission rate, which is implicitly controlled by the congestion window and the flow of TCP ACKs to maintain the virtual TCP channel efficiency to be close to the real MAC channel efficiency.

This approach proposes a new mechanism to feedback the real channel efficiency of MAC protocol to TCP in

order to alleviate the load of MAC protocol before a congestion event occurs and to enhance the network performance. Simulation results show that the proposed mechanism outperforms 802.11 DCF in terms of throughput and delay. However, this approach does not handle the effect of route failures and high BER at the intermediate nodes. Moreover, the mobility has a great impact on the channel efficiency, and it is not clear how this approach will perform over mobile networks, thus, more analysis is required.

TCPCC is a cross-layer approach conducted by Zhang et al. [37]. In this study, the authors show that the over-injection of conventional TCP window mechanism results in severe contentions, and medium contentions cause network congestion. They also show that, two important metrics, channel utilization (CU) and contention ratio (CR) should be used to characterize the network status. Then, based on these two metrics, they propose a new TCP transmission rate control mechanism based on channel utilization and contention ratio (TCPCC). In this mechanism, each node collects the information about the network busy status and determines the CU and CR accordingly. The CU and CR values fed back through ACK are ultimately determined by the bottleneck node along the flow. The TCP sender controls its transmission rate based on the feedback information.

The simulation results in [37] show that TCPCC mechanism significantly outperforms the conventional TCP mechanism and the TCP contention control mechanism in terms of throughput and end-to-end delay. Nevertheless, similar to channel efficiency-based transmission rate control, TCPCC does not have a mechanism to handle the route failures. Moreover, TCPCC requires the use of explicit feedback information from intermediate nodes. Deploying TCPCC is more difficult, since it relies on the cooperation of all nodes.

3.2.3 Comparison

Five approaches have been discussed. These approaches address the problem of TCP inability to control the traffic based on the networks conditions, which is one of the main reason for performance degradation in multi-hop wireless networks. To solve this problem, two available methods are presented. The first one is by allowing TCP to estimate the network condition without violating the layered principle. In particular, TCP has to perform some statistical operations to estimate the status and react accordingly, such as TCP-Vegas [33] and TCP-Westwood [34]. The other way is to feedback the status of the network through cross-layer information. In this way, TCP can read this information and adjust itself. The approaches that use this method are TCP-LC [35], channel efficiency-based transmission rate control [36], and TCPCC [37]. These approaches that make use of the explicit feedback from the networks provide significant improvements in TCP performance. However, if we consider the concept of protocols and applications in isolation, TCP-Westwood seems to be the preferred choice. This is because TCP-Westwood uses the idea of bandwidth estimation and can effectively alleviate the effect of the losses that are not caused by congestion. Table 2 illustrates a summary of the main characteristics of the discussed TCP enhancements.

3.3 Reducing ACK traffic overhead

From an end-to-end perspective in general and a TCP viewpoint in particular, there are two types of segment: DATA and ACKs, injected into the network by the sender and the receiver, respectively. Several researchers show that, in multi-hop wireless networks, reducing the number of segments in the communication pipe can reduce the contention and collision caused by sharing the same path. In order to achieve this, some researches proposed to enhance the overall TCP performance by reducing the number of DATA segments transmitted by the sender. Some of these enhancements are discussed in the next section. From the receiver's perspective, other researches argue that spatial contention can be reduced by introducing fewer ACK segments, say by taking advantage of their cumulative property. Such approaches aiming to reduce spatial contention caused by ACKs are named ACK-thinning, delayed ACKs, or ACK-reducing and are the point of focus for this section. These approaches have been examined in the literature and have been shown to be beneficial in terms of increasing TCP throughput.

In the legacy TCP, upon successful reception of every data packet transmitted from a sender to a receiver, an ACK response from the receiver to the sender will be generated (Figure 4a). If a data packet has not been acknowledged for some time, it is considered lost and is retransmitted by the sender. In fact, ACKs are considered control traffic used by TCP to ensure reliable data recovery. However, the small TCP ACKs consume wireless resources as much as the long TCP data packets. Moreover, the interference and collision between data and ACK packets, which are caused by sharing the same route, increase with the number of ACKs generated [39]. Therefore, it is desirable to reduce the number of control traffic (ACKs) so as to make the resources available for the actual data packets and to reduce the interference and collision between the data and the ACK packets. This is achieved by merging several ACKs in one ACK, which is possible due to the cumulative ACK option used in TCP.

The first optimization of this nature has been introduced through the standard TCP with delayed ACK option in (RFC 1122 [14], RFC 2581 [40]). With the standard delayed ACK option, TCP can generate one ACK upon receiving two in-order data packets (Figure 4b). It has been reported through extensive simulations [41,42] that TCP variants like Reno, New Reno, SACK and Vegas

Table 2 Comparison of the main characteristic of various TCP Enhancements

	"TCP-Vegas"	"TCP-Westwood"	"TCP-CL"	"Channel Efficiency-Based Transmission Rate Control"	"TCPCC"
	[33]	[34]	[35]	[36]	[37]
DEALING WITH ROUTING FAILURES	No.	No.	No.	No.	No.
DEALING WITH WIRELESS ERRORS	No.	Yes: *ssthresh* and *cwnd* are assigned based on the estimated bandwidth.	Yes: It uses negative acknowledgment (NAK) to distinguish wireless loss from congestion loss and retransmit the corrupted packet.	Yes: it uses channel efficiency to notify TCP about the available bandwidth.	Yes: channel utilization (CU) is used to characterize the network status.
DEALING WITH CONTENTION	Partial: It maintains a stabler *cwnd* which may reduce contention at lower layer.	Partial: The growth of *cwnd* is carefully controlled based on the estimated bandwidth.	Yes: it extends the IEEE 802.11 DCF scheme to reduce the effects of link-layer contention,	Partial: comparing channel efficiency to TCP rate may reduce contention at lower layer.	Yes: contention ratio (CR) is used to characterize the contention status
DEALING WITH RETRANSMISSION	No.	No.	the retransmission is based on NAK reception	No.	No.
DEALING WITH TCP CONGESTION CONTROL MECHANISM	Yes: it reduces *cwnd* only by a quarter if the loss is detected by the new faster retransmission mechanism.	Yes: *ssthresh* and *cwnd* is calculated based on the estimated bandwidth.	Yes: It modifies the slow start and *cwnd* based on NAK and ACK reception	Yes: *cwnd* is assigned based channel efficiency.	Yes: *cwnd* is assigned based on channel utilization (CU) and contention ratio (CR).
DEALING WITH HIDDEN EXPOSED PROBLEM	No.	No.	No.	No.	No.
DEALING WITH TCP RATE	Yes: the rate is estimated based on the available bandwidth	Yes: the rate is estimated based on the available bandwidth.	Yes: TCP sender controls its transmission rate based on the feedback information.	Yes: TCP sender controls its transmission rate based on the feedback information of channel efficiency.	Yes: TCP sender controls its transmission rate based on the feedback information of the network status.
DEALING WITH TCP ACK	No.	No.	Yes: the NAK notification is sent in piggyback mode with the return TCP ACK segment.	No.	No.

perform better by employing the delayed ACKs in the case of throughput, bandwidth and energy consumption. However, in 802.11 networks, the interference problem between ACK and data packets may still persist and the benefit of standard TCP with delayed ACK can be further improved. (Figure 4c) shows the case of delaying ACK for more than two packets. In fact, lowering the number of ACKs may improve TCP performance, however, the large cumulative ACKs will induce packet loss due to retransmission time out at the sender side of TCP. To address this problem, numerous enhancements and optimizations have been proposed to reduce traffic overhead caused by generating more ACKs than necessary.

3.3.1 TCP layer approaches

Dynamic delayed ACK (TCP-DDA) TCP-DDA [39] is a simple receiver modification introduced by Altman and Jiménez. In this approach, the authors investigated the impact of producing delayed ACKs for more than two received packets on TCP performance in multi-hop wireless networks. They proposed a limited dynamic delayed ACK scheme, in which the receiver begins delaying one ACK (sending one ACK for two in order packets received) and keeps increasing until four based on the sequence number of the acknowledged packet. For instance, when the delayed ACK is set to four, the receiver may send a single ACK for packet (*Pi*) implying that packets (*Pi-1*), (*Pi-2*) and (*Pi-3*) have also been successfully received, without sending separate ACKs for each. The receiver keeps delaying four ACKs except at session start-up it decreases the delayed ACK to one again.

Although the delayed ACK option decreases the frequency of RTT samples measurable by the TCP sender, this reduction of feedback has a significant impact on the RTO expiration. As a result, the TCP may retransmit all the packets that their ACKs delayed, even though the packets are received. Therefore, TCP should precisely select the optimal value of the ACK delay window. Although TCP-DDA has shown a good performance in

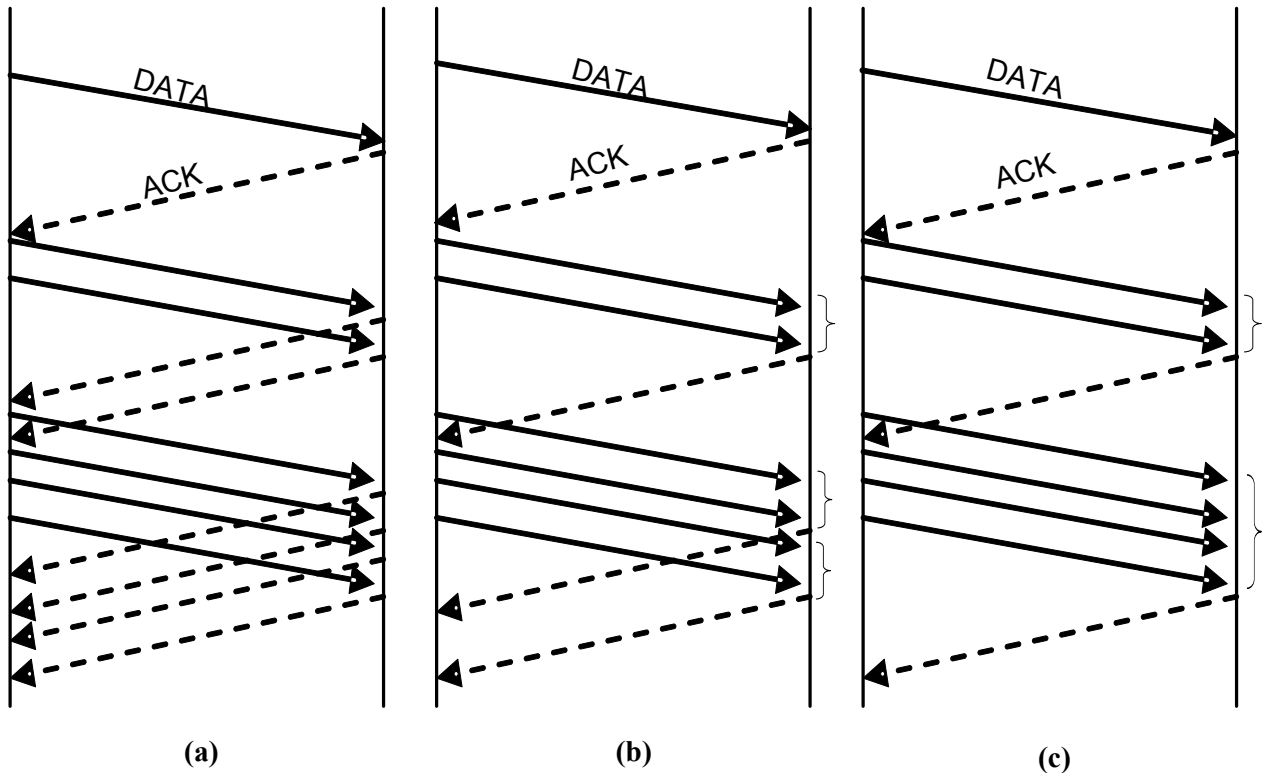

Figure 4 (a) Standard TCP, (b) standard TCP with delayed ACK option and (c) TCP with delaying more than two ACKs.

static ad hoc networks compared to the standard TCP, it may be inefficient in high traffic scenarios with considerable packet loss. Moreover, TCP-DDA has been obtained for only a single flow in static networks, multiple flows and mobility are good issues for further improvement.

Dynamic adaptive ACK (TCP-DAA) TCP-DAA [41] is a sender/receiver modification introduced by Oliveira and Braun. In TCP-DAA, the receiver reduces the number of ACKs by taking advantage of the cumulative property of TCP. TCP-DAA dictates changes to the sender as well as the receiver. In this approach, the receiver may combine up to four ACK packets when the wireless channel is in good condition and less for lossy channels. The limit of four packets is imposed by the sender's congestion window limit that is also fixed at four packets. The authors report that the low limit of the sender's *cwnd* is proper for minimizing collisions and more than enough for scenarios having up to 10 hops. The delay of ACK responses are performed in a dynamic manner based on packet loss event. When there is no packet loss, TCP-DAA delays ACKs until it receives more data packets up to four, but reduces the number to two in case of out-of-order packet delivery. These concepts lead this approach to outperform a regular TCP whenever the channel is able to provide higher bandwidth.

However, there is some processing overhead associated with this method, there is a general increase in throughput and better utilization of the wireless channel. This approach shows improvement not only in the throughput, which is the key issue in this area, but also in power consumption. Consequently, the overall energy consumption is significantly reduced, which is a key issue for battery-powered devices. TCP-DAA focuses on networks of short hops, at most ten hops, as stated in this article. Nevertheless, the evaluation includes long-hop networks, which are particularly valuable in some scenarios such as large sensor networks.

Delayed cumulative acknowledgment (TCP-DCA) TCP-DCA is a TCP receiver-based scheme introduced by Chen et al. [43]. In this scheme, the authors show that TCP does not always get throughput gain by delaying unlimited ACKs. In addition, there exists an optimal delay window size at the receiver that produces best TCP throughput. In TCP-DCA, path length is an important factor to be considered when choosing appropriate delay window sizes. Therefore, TCP-DCA selected different size of delay window to adapt TCP ACK generation based on the number of hops. When the number of hops between the sender and the receiver is less or equal to three hops, the delay window is set to be equal to the

cwnd size. For path lengths between four and nine hops, the delay window is set to be fixed with five packets. In case of long paths, more than ten hops, the receiver will set the delay window to three packets. TCP-DCA has shown good performance in multi-hop wireless networks. In conclusion, delaying ACKs for large numbers of data packets, a large delay window is always beneficial in connection with short-paths, but may be inappropriate for long-path networks. The longer the end-to-end path the longer the time for TCP sender to detect lost packets caused by delaying more ACKs. In addition, when travelling a longer path, a packet is more likely to suffer interference.

TCP-DCA is an end-to-end TCP semantic scheme, which shows good performance over static, mobile and hybrid ad hoc networks. TCP-DCA has been proposed to decrease the number of ACKs adaptively based on the path length. However, knowing the number of hops between the sender and the receiver is not a property of TCP and may require interaction with underlying layers. Moreover estimating the number of hops by TCP is not easy, especially in mobile networks where the path length is changing dynamically. Besides, TCP-DCA uses a large fixed delay window through the whole transmission between a sender and receiver and this may affect the performance with channels with high bit error rate.

TCP with Adaptive Delay Window (TCP-ADW) TCP-ADW is a TCP receiver-based modification introduced by Al-Jubari and Othman [44]. The authors show that using a fixed delay window is invalidated in this changing environment and the optimal delay window size at the receiver that produces best TCP throughput can be achieved dynamically. TCP-ADW adjusts the delay window dynamically based on several conditions such as transmission rate, slow start phase, path length and packet lost event. To reduce the number of ACK packets appropriately, TCP-ADW makes the ACK generated by reaching the optimal dynamic delay window. In this way, the receiver will be able to adapt itself to different values of delays imposed by the wireless channel conditions. In addition, it delays just enough to avoid the transmission time-out at the sender. In this approach, unless the sender's retransmission timer expires, the receiver always increases the delay window based on the increase in the transmission rate, except at session startup. During the startup, the receiver sets the delay window to one and increases it based on the transmission rate. When the packet loss occurs, the receiver decreases its delay window to a certain value based on the hop count (path length). In case of short path, the receiver decreases the delay window to half. For the long path, however, the receiver will decrease its delay window to two. This is because the TCP sender will take a long time to detect lost packets obtained by delaying more ACKs. Out-of-order packets immediately cause the ACK generation to inform the sender of the packet loss/recovery in a timely manner, as introduced in the recommendation of RFC 2581 [40]. The receiver uses a fixed interval of 200 ms for timing out.

TCP-ADW is an end-to-end TCP solution, which shows a significant performance over TCP-DCA and much more over the regular TCP in static ad hoc networks. However, evaluating TCP-ADW over mobile ad hoc networks and studying its performance over channels with high BER is required.

3.3.2 Cross-layer approaches

Monitoring Delayed Acknowledgment (TCP-MDA) MDA is across-layer-based modification introduced by Armaghani et al. [45]. TCP-MDA proposed a dynamic interaction strategy between TCP and MAC layers, which reduces the number of ACKs by monitoring the channel condition. To properly set the number of delayed ACKs in TCP, TCP-MDA uses a mechanism to collect collision probability along the path from sender to receiver in MAC layer. Based on the estimated collision probability, TCP adjusts itself to the channel condition by delaying less ACKs in high traffic conditions and more ACKs in low traffic conditions. If the channel is in a good condition, when the estimated collision probability is less than a predefined threshold, the TCP receiver may combine up to four ACKs. However, when a high collision in the MAC layer is observed, the TCP receiver generates more ACKs to avoid unnecessary retransmission caused by time-out in the sender side. The limit of four ACKs is imposed by the sender's *cwnd*, which is defined as the maximum number of data packets a TCP sender may inject into the network at any time without waiting for an ACK from the receiver. This low limit of four packets for *cwnd* is proper to minimize the channel contention in short range scenarios [46].

The simulation results show a throughput improvement over TCP-DAA and much more over the regular TCP in different scenarios. TCP-MDA shows that the optimal number of delayed ACKs is based on the path length of a TCP connection and a large delay window may solely improve TCP. In addition, the results show that an optimal number of delayed ACKs exists, which produces the best throughput in different ranges. Therefore, TCP-MDA shows that a large optimal number can solely improve TCP throughput in short scenarios of less than five hops. However, a longer path congestion window limit provides more throughput gain. However, as TCP-MDA has been designed to perform well in static networks, evaluating TCP-MDA over mobile networks is required. In addition, TCP-MDA requires intermediate nodes to notify TCP about channel conditions and this makes its deployment and implementation complicated. Moreover, limiting the *cwnd* to four may affect the

performance of TCP in case of high transmutation rate and low BER.

3.3.3 Comparison

Five approaches have been presented. These approaches address the problem of reducing the overhead caused by ACKs traffic on the wireless channel, which is one of the key issues of TCP performance degradation in multi-hop wireless networks. Some approaches are based on end-to-end TCP layer modification, such as DDA, DAA, DCA, ADW and others are based on cross-layers, such as MDA. DDA and DAA reduce the number of ACKs dynamically but it is limited up to four packets. DDA increases the delay window based on the increase of sequence number of packets. However, DAA increases the delay window based on in-order packets delivery and reduces the delay window based on the loss events (out-of-order packet delivery). For the DCA approach, the delay window is adaptively set based on the number of hops. DCA shows a good performance over DAA, which, in turn, outperform DDA. In ADW, the delay window is not limited and is set adaptively and dynamically based on three impotent factors, number of hops, transmission rate, and loss event. The results of ADW show that ADW has a significant performance over DCA and much more over the standard TCP in static ad hoc networks. In the cross-layer approach, use the feedback from the MAC layer to set the delay window. MDA uses a limited delay window of up to four packets to generate an ACK. The simulation results of MDA illustrate that MDA outperform DAA and the regular TCP as well. Table 3 illustrates a summary of the main characteristics of the discussed TCP enhancements.

3.4 Limiting TCP aggressiveness

It is well known that one of the main functions of TCP is the TCP window mechanism that controls the amount of traffic sent into the network. However, one of the critical reasons for poor TCP performance in multi-hop wireless networks lies in the aggressive window increase policy of TCP itself. Thus, several TCP enhancements have been proposed recently to address the problem of TCP aggressiveness. In Section 2.4, we discussed the adverse effects of the inter-flow and intra-flow contention problems on the TCP performance. The reason for this contention is that conventional TCP does not operate its congestion window at an appropriate level. The representative schemes that reduce contention by limiting TCP aggressiveness include "TCP-Vegas-W" [47], "Congestion Window Limit" [24], "fractional window increment (FeW)" [48], "Adaptive packet size on top of FeW (APS-FeW) [49] and "Link RED and Adaptive Pacing" [46].

3.4.1 TCP layer approaches

Fractional window increment (FeW) FeW is a cross-layer approach proposed by Nahm et al. [48]. In this study, the authors investigate the effect of congestion and MAC contention on the interaction between TCP and on-demand ad hoc routing protocols in the 802.11 ad hoc networks. They have observed that TCP generally operates at a high rate and induces the overreaction of routing protocol. Therefore, limiting this aggressiveness of TCP congestion window may dramatically improve the quality of end-to-end connection. To this end, the authors propose a fractional window increment (FeW) scheme in which, TCP uses a fractional window update instead of using exponential window update associated with conventional TCP. At each ACK reception, the TCP sender updates *cwnd* by Equation 1. In this way, FeW will force TCP to function with a very small fractional rate ($0 < \alpha \leq 1$) at every RTT:

$$cwnd^{new} = cwnd^{current} + \frac{\alpha}{cwnd^{current}} \quad (1)$$

The simulation results in [48] demonstrate that FeW improves TCP performance dramatically, which verifies FeW's assumption that its window prediction mechanism is still as accurate as that in legacy TCP. The wireless connection can benefit from both the quick reaction of legacy TCP and the load alleviation of FeW. However, it is not yet clear to what extent short connections with only relatively small amounts of data might suffer from the slower congestion window growth and the resulting slower convergence. Moreover, the window update mechanism of FeW cannot make full use of its accurate predicted window for transmission, because, in practice, this leads to a scheme with a non-integer increment in the window size per RTT. Although FeW adjusts its congestion window in the same pattern as legacy TCP does, the fractional part of its window takes no effect on transmission and a certain amount of predicted network capacity is wasted. A solution for this problem is provided in [49].

APS-FeW Wang et al. proposed a new adaptive packet size (APS) [49] enhancement of FeW [48]. Both FeW and APS-FeWare based on the observation that TCP induces the over-action of routing protocol and reduces the performance of the connection. As shown in [48], the FeW scheme improves the connection performance by limiting TCP's aggressiveness. The authors in [49] show that, to some extent, FeW is too strict in that it eliminates the possibility of delivering more bytes under the same congestion window. Based on their study on the window update mechanism of FeW, they found that FeW cannot make full use of its accurate predicted window for transmission. Although FeW adjusts its congestion window in the same pattern as legacy TCP does, the fractional part of its window may waste the network capacity. To solve this problem, [49] proposes an adaptive packet size (APS) scheme to work on top of FeW for TCP. Since the packet

Table 3 Comparison of the main characteristic of various TCP Enhancements

	" TCP-DDA " [39]	" TCP-DAA " [41]	" TCP-DCA " [43]	" TCP-ADW " [44]	" TCP-MDA " [45]
DEALING WITH ROUTING FAILURES	No.	No.	No.	No.	No.
DEALING WITH WIRELESS ERRORS	No.	No.	No.	No.	Partially: the setting of delayed ACK is based on the information collected from the channel; this may reduce the effect of wireless errors.
DEALING WITH CONTENTION	Yes: reducing the number of ACK may reduce the effect of contention.	Yes: reducing the number of ACK may reduce the effect of contention.	Yes: reducing the number of ACK may reduce the effect of contention.	Yes: reducing the number of ACK may reduce the effect of contention.	Yes: reducing the number of ACK may reduce the effect of contention.
DEALING WITH RETRANSMISSION	No.	No.	No.	No.	No.
DEALING WITH TCP CONGESTION CONTROL MECHANISM	No.	Yes: *cwnd* is limited to maximum of four packets.	No.	No.	Yes: *cwnd* is limited to maximum of four packets.
DEALING WITH HIDDEN EXPOSED PROBLEM	No.	No.	No.	No.	No.
DEALING WITH TCP RATE	No.	Yes: the rate of TCP is limited due to *cwnd* limitation.	No.	No.	Yes: the rate of TCP is limited due to *cwnd* limitation.
DEALING WITH TCP ACK	Yes: it delays the ACK for a maximum of four packets. At the start up, it reduces the delayed ACK to one.	Yes: it delays the ACK for a maximum of four packets. In the case of packet lost, it reduces the delayed ACK to one.	Yes: it delays the ACK adaptively based on the number of hops.	Yes: it delays the ACK dynamically based on the transmission rate, loss event, and number of hops.	Yes: it delays the ACK based on the feedback from the MAC layer.

size in the start phase is an integer value, APS performs exactly the same as TCP does. When the congestion window exceeds the threshold, *cwnd* becomes a fractional number. Unlike FeW, which has a fixed packet size, APS on top of FeW (APS-FeW) can adapt the packet size to current predicted window and make full use of the window for transmission. It defines the initial packet size (*initPacketsize*) as the fixed packet size when TCP resets, and uses the (*cwnd*) to calculate the current packet size, as in the following Equation 2:

$$\text{Packetsize} = \left\lfloor \frac{cwnd \times initPacketsize}{\lfloor cwnd \rfloor} \right\rfloor \quad (2)$$

The procedure of APS-FeW is as follows. (1) When TCP source receives an ACK it updates its congestion window and current packet size, as shown in Equation 2. Then, it keeps using this packet size to transmit the following packets until the next ACK arrives. (2) When the retransmit timer is out, the TCP enters slow start, the congestion window resets to 1. The TCP source needs to repack the data in its buffer with initial packet size and retransmit it. (3) When TCP enters quick start due to three duplicated ACKs, the TCP source does not need to repack the lost packet, it just retransmits the packet in its buffer.

The proposed scheme utilizes the advantages of both legacy TCP and FeW to improve TCP performance over multi-hop 802.11 networks. Through extensive simulation results presented in [49], this approach shows that APS over FeW outperforms FeW. With APS-FeW, the wireless connection can benefit from both the quick reaction of legacy TCP and the load alleviation of FeW.

TCP-Vegas-W In [47], Ding et al. proposed Vegas-W, which is a modified TCP protocol based on TCP-Vegas [33] for multi-hop ad hoc networks. This approach is based on the observation that TCP Vegas cannot maintain the optimal window with maximum average throughput when the network capacity is smaller than the reset slow start threshold of Vegas. Based on the authors study, this problem stems from the large minimum congestion window of Vegas, large reset slow start threshold and aggressive window increase policy. All of

them induce overload of the network, which causes packet losses at MAC layer, over-reaction at routing layer and, consequently, the aggregate throughput of all traffic will decrease. To solve this problem, the authors propose Vegas-W, in which the congestion window is extended a fraction with a rate control timer under the TCP sending process. In TCP-Vegas-W, the probing mechanisms of legacy TCP-Vegas in both slow start and congestion avoidance phases have been changed to increase the congestion window after receiving more than one ACK. In addition, the slow start threshold is modified to be updated by tracking stable window.

This approach considers the special features of wireless ad hoc networks and improves legacy Vegas. The simulation results show that Vegas-W improves throughput of Vegas significantly over a wide variety of scenarios. They also show that Vegas-W obtains higher throughput than Vegas and FeW. However, in analysing the interactions between TCP and the underling protocols, routing and MAC protocols, over multi-hop wireless circumstances are required for further improvement.

3.4.2 Link layer approaches

Link RED and adaptive pacing LINK RED and AP [46] are techniques proposed by Fu et al. that allow TCP to react pre-emptively to link overload by adaptively delaying certain packet transmissions to reduce contention. The authors in this article show that a small TCP congestion window can have beneficial effects on the TCP performance in mobile ad hoc. This approach has shown that a maximum of 1/4 spatial reuse improves TCP performance for a one-way flow in certain scenarios. This study implies that limiting the maximum sending rate of a TCP source may alleviate the congestion window overshoot problem, which, in turn, may reduce the contention at the MAC layer. In the following section, we briefly explain the two proposed techniques used in this approach. *link random early detection (Link RED)* [46] is a link layer active queue management algorithm that exploits explicit congestion notification (ECN) marking to stabilize TCP window. Link RED aims to reduce the contention on the wireless channel. This is done by maintaining an average number of retries for recent packet transmissions. If the average retry attempt value exceeds a given threshold value, Link RED will mark the outgoing packets with a probability depending on the value, computed according to the RED algorithm [50]. The authors suggested increasing the back-off time at the MAC layer, TCP will then reduce its sending rate and thereby, to some extent, avoid packet loss.

Adaptive Pacing (AP) [46] aims to improve spatial channel reuse and this is done by distributing traffic among intermediate nodes in a more balanced way. In the current IEEE 802.11 protocol, a node is constrained from contending for the channel by a random back-off period, plus a single packet transmission time that is announced by the RTS or CTS frame. However, the contention related drops caused by the exposed receivers problem persists due to the lack of coordination between nodes that are two hops away from each other. To solve this problem, AP lets some nodes wait, in addition to the normal back-off period, for an extra amount of time equal to a packet transmission time when necessary. AP is used in coordination with Link RED as follows. When a node finds its average number of transmission retries to be less than a threshold, it calculates its back-off time as usual. When the average number of retries goes beyond this threshold, Link RED will start to mark packets and AP will then increase the back-off time of the pending transmission by an interval equal to the transmission time of the previous packet. Link RED provides an early sign of network overload, which helps TCP improve the interflow fairness between multiple TCP sessions. When Link RED is used in conjunction with AP, they improve the spatial reuse by reducing contention and, thus, improve the TCP performance. However, in this approach, the additional back-off time is based on the packet size, therefore, existence of different data packets size in the network should be inspected. Moreover, Link RED requires the MAC layer to maintain an average transmission retry attempt value. It also requires a RED-like algorithm to be implemented at the MAC layer, and this complicates the implementation and the deployment of the scheme.

3.4.3 Cross-layer approaches

Congestion window limit (CWL) Chen et al. designed CWL [24] to reduce the contention at the MAC layer in order to improve TCP performance. The main goal of this approach is to adjust the maximum congestion window size dynamically based on the current path length. Therefore, the sending rate will not exceed the maximum spatial reuse of the channel. By this, both the intra-flow and interflow contention problems are reduced. The authors in [24] turn the problem of setting proper CWL into identifying the bandwidth-delay product (BDP) of the path. Based on this methodology, they first show and prove that, independent of any specific MAC layer protocol, the upper bound of a path's BDP cannot exceed its round-trip hop-count (RTHC). Therefore, CWL must be used with routing protocols that are aware of the path length. By considering the transmission interference of the IEEE 802.11 MAC layer protocol, this approach derives a tighter upper bound of BDP, which is approximately 1/5 of the RTHC in a certain topology. Based on this tighter bound, it is shown that TCP throughput can be improved by setting its CWL dynamically according to the current RTHC of the path. This approximation of 1/5 can be easily explained by also considering an increase in contention (and, thus, a reduced spatial reuse) contributed by the reverse ACK packets along the same path. For the IEEE 802.11 MAC, this approach gives an even tighter bound of RTHC 5. DSR

routing protocol is used as a path-aware routing protocol to get information about the path length at the source node. This allows for setting the CWL dynamically depending on the path length of the connection. CWL is a dynamic congestion window limit based on the broadcast characteristics of the wireless medium. In this approach, based on the tighter bound approximation, 1/5 of RTHC, the window limit is set for only a single flow. However, for multiple flows that compete, it is unclear that this will always hold. The factor could depend on density and the number of competing connections. In addition, no comprehensive study has been given when using a mobile network, where the length of path is changing dynamically. Moreover, in wireless multi-hop communications, competition for bandwidth between DATA and ACK segments is even more pronounced due to the broadcasting nature of the shared wireless medium and the limited available bandwidth, however, no comprehensive study has been given undertaken. To demonstrate the performance gain of their scheme, different simulations were conducted to compare it to TCP Reno with an unbounded congestion window. Nevertheless, it should be noted that they also changed the maximum retransmission time-out of TCP in their simulations in order to let TCP probe the route quickly, setting it to 2 s as opposed to the 240 s given in RFC 1122 [14]. This might affect the simulation results.

3.4.4 Comparison

Five approaches have been presented. These approaches address the problem of TCP overload, which is one of the key factors of the contention problem in multi-hop wireless networks. These approaches are classified to three sets based on the modification layer, which are TCP layer approaches, network layer approaches and cross-layer approaches. Recent research argues that the network overload is the major factor contributing to contention and packet losses in multi-hop wireless networks [48,49]. With a fine tuned TCP window mechanism, the network load can be kept at a reasonable level, and, consequently, the performance of TCP is improved. However, other research proposed to solve this problem from its origin at the link layer [46]. While others attribute this problem to the lack of coordination between TCP and the other layers and suggest that the cross-layering is the best solution. In FeW [48], the mean goal is to force TCP to function with a very small fractional rate. The evaluation of FeW scheme shows that FeW outperforms Link RED [46]. However, the fractional part of its window may waste the network capacity. ASP-FEW [49] proposed solving this problem by dynamically changing the packet size beads on the congestion window size. APS-FeW shows a good performance over FeW in terms of throughput. Vegas-W [47] also shows an improved TCP performance over FeW. This approach has been proposed to overcome the problems of the large minimum congestion window, large reset slow start threshold and aggressive window increase policy of TCP-Vegas in multi-hop wireless networks. In the link layer proposals, Link RED and Adaptive Pacing proposed to reduce the contention problem from its origin at the link layer. The main idea behind this approach is to control the TCP window size by tuning up the link layer dropping probability according to perceived channel contention. CWL [24] is a simple cross-layer approach that aims to reduce the contention on wireless channel by limiting the TCP congestion window based on identifying the round-trip hop-count of the path. However, it must be used with a routing protocol that is aware of the path length. Table 4 illustrates a summary of the main characteristics of the discussed TCP enhancements.

4 Further discussion and open research issues

In this section, a discussion on the TCP issues in multi-hop wireless networks is provided. In particular, the discussion addresses the following questions that arise. Which TCP enhancement seems to be the best for multi-hop ad hoc networks? Are the current TCP enhancements sufficient to solve the problem of TCP in such environments? It is well known that the main factors affecting TCP performance in wireless networks are the underlying layer characteristics, such as route failures and wireless channel errors. Therefore, those schemes that make use of explicit feedback from intermediate relay nodes to detect route failures or congestion status almost always provide significant improvements in TCP performance. TCP with cross-layer solutions have attracted much attention recently and they report better performance than layered solutions. The obvious benefit of using the cross-layer-based approach is that it is more accurate because the information is directly from the network. Many cross-layer solutions have been proposed in this article to deal with typical TCP problems in multi-hop wireless links, such as wireless losses, drastic changes in routes, or bandwidth availability. These approaches make TCP aware of what is happening at the underlying layers and modify its behaviour to react according to network conditions, thus, improving its performance. More importantly, such schemes are able to manage and utilize the available bandwidth in the network more efficiently. However, in terms of complexity, cross-layer solutions are more complex and more difficult to implement than layered solutions. In addition, they break the concept of designing protocols to be independent of other layer protocols. The design of cross-layers requires a consideration of the system in its entirety. Therefore, to choose between cross-layer and layered solutions, we first have to answer: what is the priority for us? performance optimization or deployment cost. By considering TCP enhancements in respect of the concept of designing protocols (with minor modifications to conventional TCP or to

Table 4 Comparison of the main characteristic of various TCP Enhancements

	"FeW" [48]	"APS-FeW" [49]	"Vegas-W" [47]	"Link RED and AP" [46]	"CWL" [24]
DEALING WITH ROUTING FAILURES	No.	No.	No.	No.	No.
DEALING WITH WIRELESS ERRORS	No.	No.	No.	No.	No.
DEALING WITH CONTENTION	Yes: it reduces the TCP window over-shoot to reduce the contention.	Yes: it reduces the TCP window over-shoot to reduce the contention.	Yes: can be reduced by controlling the rate.	Yes: it detects early sign of congestion. The sending rate is reduced to improve spatial reuse and alleviate the contention.	Yes: contention can be reduced by limiting the *cwnd*.
DEALING WITH RETRANSMISSION	No.	No.	No.	No.	Yes: the maximum retransmission time-out of TCP is modified to 2 second instead of 240 second.
DEALING WITH TCP CONGESTION CONTROL MECHANISM	Yes: the *cwnd* is updated by fractional window.	Yes: it uses byte count instead of packet count to calculates *cwnd*. The packet size varies based on *cwnd* size.	Yes: both slow start and congestion avoidance mechanisms in TCP-Vegas are modified to increase *cwnd* after receiving more than one ACK. It also updates slow start threshold by tracking stable window.	No.	Yes: limiting the congestion window by the round-trip-hop-count.
DEALING WITH HIDDEN EXPOSED PROBLEM	No.	No.	No.	Not directly but the problem can be reduced by considering spatial channel reuse.	Not directly but the problem can be reduced by considering spatial channel reuse.
DEALING WITH TCP RATE	Yes: low rate caused by small window.	Yes: low rate caused by small window.	Yes: slow rate cased by congestion window fractions and rate control timer.	Yes: slow rate caused by (ECN) marking.	Yes: low rate caused by limited congestion window.
DEALING WITH TCP ACK	No.	No.	No.	No.	No.

the end nodes), layered-based schemes seem to be the preferred choice. In fact, an ultimate solution for solving all types of wireless network problems is unlikely to be developed. Even though numerous TCP enhancements have been proposed to improve the performance of TCP in multi-hop wireless networks, there does not exist an approach that promises the best and complete solution. Most wireless TCP enhancements tend to tackle a specific and limited set of problems that arise. Therefore, they are insufficient to provide the ultimate solution. It may be worthwhile to combine various approaches together in order to provide a much more comprehensive solution. However, doing this requires a comprehensive analysis of their assumptions and their relative strengths and weaknesses. Moreover, it is necessary to understand each solution's properties and suitable application scenarios. Thus, it is difficult to make an overall conclusive evaluation of one solution vs. another. Furthermore, the incompatibility of these schemes with each other is another challenge. In Table 5 a summary of the TCP enhancements listed in this article and their targeted features is presented. The comparison is based on five features: the solution type column shows whether the approach is based on a cross-layer or layered solution. The enhancements that preserve the end-to-end semantic of TCP are listed. For the ease of deployment, the approaches that require a modification at the intermediate nodes are considered difficult and those solutions in the end nodes are identified as easy for development. The degree of complexity can be high, medium or low. Layered solutions are considered to be medium or low complexity, whereas cross-layer solutions are considered to be medium or high complexity. The targeted network that an approach was designed for and the evaluation method are also presented in the table.

Although the existing algorithms provide some possible solutions to alleviate the problems of TCP in wireless networks, some issues have not been discussed in the literature and are potential research topics. In the following, we show some of the issues and questions that may give some directions for future research. The frequent route failures and route re-establishments in multi-hop wireless networks introduce a new challenge for TCP

Table 5 Comparison of the main characteristic of various TCP Enhancements

StrategyApproach		Solution Type	TCP Semantic	Ease Of Deployment	Evolution Method	Complexity Degree	Targeted Network	TCP Connections	Layering Violation
Determining routing failures	"Fixed RTO"	TCP: Sender	End-To-End	Easy	Simulation	Low	Mobile	Multiple	No
	"TCP DOOR"	TCP: Sender/ Receiver	End-To-End	Easy	Simulation	Low	Mobile	Single	No
	"Backup Path Routing"	Network layer	End-To-End	Easy	Simulation	Low	Mobile	Single	No
	"Routing Exploiting Multiple Interfaces"	Network layer	End-To-End	Difficult	Simulation	Medium	Static/Mobile	Multiple	No
	"ELFN"	Cross-layer: TCP/ Network	End-To-End	Easy	Simulation	Medium	Static/Mobile	Multiple	Yes
Estimating bandwidth and channel status	"TCP-Vegas"	TCP Sender	End-To-End	Easy	Simulation/ Measurement	Low	Wired	Multiple	No
	"TCP-Westwood"	TCP: sender	End-To-End	Easy	Simulation	Low	Mixed wired/ wireless	Multiple	No
	"TCP-CL"	Cross-layer: TCP/ Data Link	End-To-End	Difficult	Simulation	High	Static	Multiple	Yes
	"Channel Efficiency-Based Transmission Rate Control"	Cross-layer: TCP/ Data Link	End-To-End	Difficult	Simulation	Medium	Static	Multiple	Yes
	"TCPCC"	Cross-layer: TCP/ Data Link	End-To-End	Difficult	Simulation	High	Static	Multiple	Yes
Reducing ACK overhead	"TCP-DDA"	TCP receiver	End-To-End	Easy	Simulation	Low	Static	Single	No
	"TCP-DAA"	TCP: sender/ receiver	End-To-End	Easy	Simulation	Low	Static	Single	No
	"TCP-DCA"	TCP: receiver	End-To-End	Easy	Simulation/ Measurement	Low	Static/mobile	Multiple	No
	"TCP-ADW"	TCP: receiver	End-To-End	Easy	Simulation	Low	Static	Multiple	No
	"TCP-MDA"	Cross-layer: TCP/ Data Link	End-To-End	Difficult	Simulation	High	Static	Multiple	Yes
Limiting TCP Aggressiveness	"FeW"	TCP sender	End-To-End	Easy	Simulation	Low	Static/Mobile	Multiple	No
	"APS-FeW"	TCP: sender	End-To-End	Easy	Simulation	Low	Static/Mobile	Multiple	No
	"Vegas-W"	TCP: sender	End-To-End	Easy	Simulation	Low	Static	Multiple	No
	"Link RED and AP"	Link Layer	End-To-End	Easy	Simulation	Medium	Static	Multiple	No
	"CWL"	Cross-layer: TCP/ Network	End-To-End	Easy	Simulation	Low	Static	Multiple	Yes

congestion control algorithms. Is the function that is used in TCP and its enhancements to compute the retransmission time-out after a route re-establishment efficient and valid in multi-hop wireless networks? Do the current congestion control algorithms of TCP and its enhancements behave efficiently in dynamic environments with mobility? Besides, the current solution provides no specific mechanisms to avoid burst loss due to temporary disconnections. They are therefore inadequate at alleviating performance problems due to burst losses and temporary disconnections resulting from mobility. Moreover, the scalability issue of 802.11 and its amendments such as 802.11n has not been well addressed. Will the enhancements and the emergence of new standards of wireless solve the problem of TCP? How does TCP behave when applied over these new technologies? In addition, the interaction of increasing initial window size with delay spike has not been considered. From the cross-layer perspective and to the best of our knowledge, there is no comprehensive study that provides a good solution to the interaction problem between traffic congestion and MAC contention in such environments. In summary, some issues are needed to be considered to resolve the problems for TCP in wireless networks. We believe the following is desirable in a good TCP algorithm in wireless networks:

Minimize the occurrence of packets loss and minimize packet retransmissions.

Operate as compatible sender-receiver algorithms in order to yield high integration and interoperability.

Provide efficient estimation of the bandwidth in order to achieve high connection throughput.

Have the ability to distinguish between loss types and response accordingly and rapidly.

Avoid retransmissions by time-outs.

Minimize excessive ACKs and avoid injecting traffic bursts into the networks.

Achieve low algorithm complexity.

5 Conclusion

TCP is the most used reliable transport protocol in the Internet. This makes it a natural choice for reliable data delivery in multi-hop wireless networks. This article provides an extensive survey, which summarizes the problems of using TCP over multi-hop wireless networks. It is clear that the most critical problem facing TCP when applied over multi-hop wireless networks is its inability to cope with the effects of route failure and wireless errors. TCP assumes that losses are always due to network congestion. While this assumption in most cases is true in wired networks, it is, unfortunately, not valid in wireless networks. Several major TCP enhancements have been made in the literature to remedy or alleviate these problems. However, they introduce significant improvements over conventional TCP, most of them tend to tackle a specific or limited set of common wireless problems. In summary, existing TCP enhancements are not sufficient in isolation to provide a complete reliable TCP solution. Therefore, designing a suitable solution to improve TCP performance in multi-hop wireless networks is still a task for future research.

Acknowledgements

This study was supported by the Research University Grant Scheme (RUGS Number: 05/03/10/1039RU).

Author details

[1]Department of Communication Technology and Network Faculty of Computer Science and Information Technology, Universiti Putra Malaysia 43400 UPM, Serdang, Selangor DE, Malaysia [2]Department of Computer and Communication System Engineering Faculty of Engineering, Universiti Putra Malaysia 43400 UPM, Serdang, Selangor DE, Malaysia

Competing interests

The authors declare that they have no competing interests.

Received: 30 December 2010 Accepted: 7 December 2011
Published: 7 December 2011

References

1. Postel J (ed.), Transmission control protocol RFC 793, IETF Network Working Group, (October 1981)
2. M Patel, N Tanna, P Patel, R Banerjee, TCP over Wireless Networks: Issues, Challenges and Survey of Solutions. (November 2001) http://citeseer.ist.psu.edu/489782.html
3. K Pentikousis, TCP in wired-cum-wireless environments. IEEE Commun Surveys Tutorials. **3**(4), 2–14 (200)
4. A Al Hanbali, E Altman, P Nain, A survey of TCP over ad hoc networks. IEEE Commun Surveys Tutorials. **7**(3), 22–36 (2005). doi:10.1109/COMST.2005.1610548
5. X Chen, H Zhai, J Wang, Y Fang, A survey on improving TCP performance over wireless networks, in *Resource Management in Wireless Networking, ser. Network Theory and Applications*, vol. 16, ed. by Cardei M, Cardei I, Du D-Z (Springer, New York, 2005), pp. 657–695. doi:10.1007/0-387-23808-5_23
6. A survey on TCP over mobile ad-hoc networks Nova Science Publishers, (2005)
7. B Sardar, D Saha, A survey of TCP enhancements for last-hop wireless networks. IEEE Commun Surveys Tutorials. **8**(3), 20–34 (2006)
8. S Thangam, E Kirubakaran, A survey on cross-layer based approach for improving TCP performance in multi hop mobile ad hoc networks. in *International Conference on Education Technology and Computer, ICETC '09* 294–298 (April 2009)
9. M Allman, V Paxson, E Blanton, TCP congestion control, RFC 5681, IETF Network Working Group (September 2009)
10. V Paxson, M Allman, Computing TCP's retransmission timer, (RFC 2988, IETF Network Working Group, November 2000)
11. W Stevens, TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms, RFC 2001, IETF Network Working Group (November 1997)
12. A Roach, A negative acknowledgement mechanism for signaling compression, RFC 4077, IETF Network Working Group (May 2005)
13. M Mathis, J Mahdavi, S Floyd, A Romanow, TCP selective acknowledgment options, RFC 2018, IETF Network Working Group (October 1996)
14. R Braden, Requirements for internet hosts–communication layers, RFC 1122, IETF Network Working Group (October 1989)
15. P Bhagwat, P Bhattacharya, A Krishna, S Tripathi, Using channel state dependent packet scheduling to improve TCP throughput over wireless LANs. Wirel Netw. **3**, 91–102 (1997). doi:10.1023/A:1019132612232
16. A Chockalingam, M Zorzi, V Tralli, Wireless TCP performance with link layer FEC/ARQ, in *Proc IEEE International Conference on Communications (ICC '99)*. **2**, 1212–1216 (1999)

17. F Tobagi, L Kleinrock, Packet switching in radio channels: Part ii–the hidden terminal problem in carrier sense multiple-access and the busy-tone solution. IEEE Trans Commun. **23**(12), 1417–1433 (1975). doi:10.1109/TCOM.1975.1092767
18. V Bharghavan, A Demers, S Shenker, L Zhang, Macaw: a media access protocol for wireless LAN's. SIGCOMM Comput Commun Rev. **24**, 212–225 (1994). doi:10.1145/190809.190334
19. IEEE 802.11 WLAN standard http://standards.ieee.org/getieee802
20. Z Fu, P Zerfos, H Luo, S Lu, L Zhang, M Gerla, The impact of multihop wireless channel on TCP throughput and loss. in *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications (INFOCOM 2003) IEEE Societies*. **3**, 1744–1753 (March 2003)
21. K Xu, M Gerla, S Bae, Effectiveness of RTS/CTS handshake in IEEE 802.11 based ad hoc networks. Ad Hoc Netw. **1**(1), 107–123 (2003). doi:10.1016/S1570-8705(03)00015-5
22. D Berger, Z Ye, P Sinha, S Krishnamurthy, M Faloutsos, S Tripathi, TCP-friendly medium access control for ad-hoc wireless networks: alleviating self-contention. in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems* 214–223 (October 2004)
23. H Zhai, X Chen, Y Fang, Alleviating intra-flow and inter-flow contentions for reliable service in mobile ad hoc networks. in *IEEE Military Communications Conference (MILCOM 2004)*. **3**, 1640–1646 (2004)
24. K Chen, Y Xue, K Nahrstedt, On setting TCP's congestion window limit in mobile ad hoc networks. in *IEEE International Conference on Communications (ICC '03)*. **2**, 1080–1084 (2003)
25. R Ludwig, RH Katz, The Eifel algorithm: making TCP robust against spurious retransmissions. SIGCOMM Comput Commun Rev. **30**, 30–36 (200)
26. TD Dyer, RV Boppana, A comparison of TCP performance over three routing protocols for mobile ad hoc networks. in *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking and Computing. ser. MobiHoc '01* 56–66 (2001)
27. F Wang, Y Zhang, Improving TCP performance over mobile ad-hoc networks with out-of-order detection and response. in *Proceedings of the 3rd ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '02)* 217–225 (2002)
28. CE Perkins, Ad Hoc Networking (Addison-Wesley (E) New York, 2001)
29. H Lim, K Xu, M Gerla, TCP performance over multipath routing in mobile ad hoc networks. in *IEEE International Conference on Communications (ICC '03)*. **2**, 1064–1068 (2003)
30. S-J Lee, M Gerla, Split multipath routing with maximally disjoint paths in ad hoc networks. in *IEEE International Conference on Communications (ICC 2001)*. **10**, 3201–3205 (2001)
31. W Yoon, N Vaidya, Routing exploiting multiple heterogeneous wireless interfaces: A TCP performance study. Comput Commun. **33**(1), 23–34 (2010). doi:10.1016/j.comcom.2009.07.012
32. G Holland, N Vaidya, Analysis of TCP performance over mobile ad hoc networks. Wirel Netw. **8**, 275–288 (2002). doi:10.1023/A:1013798127590
33. L Brakmo, S O'Malley, L Peterson, TCP vegas: new techniques for congestion detection and avoidance. ACM SIGCOMM Comput Commun Rev. **24**(4), 24–35 (1994). doi:10.1145/190809.190317
34. C Casetti, M Gerla, S Mascolo, TCP Westwood: End-to-end bandwidth estimation for enhanced transport over wireless links. Wirel Netw. **8**(5), 467–479 (2002). doi:10.1023/A:1016590112381
35. R-S Cheng, H-T Lin, A cross-layer design for TCP end-to-end performance improvement in multi-hop wireless networks. Comput Commun. **31**(14), 3145–3152 (2008). doi:10.1016/j.comcom.2008.04.017
36. X Zhang, J Lv, X Han, DK Sung, Channel efficiency-based transmission rate control for congestion avoidance in wireless ad hoc networks. IEEE Commun Lett. **13**(9), 706–708 (2009)
37. X Zhang, N Li, W Zhu, D Sung, TCP transmission rate control mechanism based on channel utilization and contention ratio in ad hoc networks. IEEE Commun Lett. **13**(4), 280–282 (2009)
38. S Xu, T Saadawi, Performance evaluation of TCP algorithms in multi-hop wireless packet networks. Wirel Commun Mob Comput. **2**(1), 85–100 (2002). doi:10.1002/wcm.35
39. E Altman, T Jiménez, Novel delayed ACK techniques for improving TCP performance in multihop wireless networks, in *Proceedings of Personal Wireless Communications (PWC03)*, vol. 3. (Venice, Italy, 2003), pp. 237–250
40. M Allman, V Paxson, W Stevens, TCP congestion control, (RFC 2581, IETF Network Working Group, April 1999)
41. R Oliveira, T Braun, A Smart TCP acknowledgment approach for multihop wireless networks. IEEE Trans Mobile Comput. **6**(2), 192–205 (2007)
42. W Lilakiatsakun, A Seneviratne, TCP performances over wireless link deploying delayed ACK. in *The 57th IEEE Semiannual Vehicular Technology Conference (VTC 2003, Spring)*. **3**, 1715–1719 (2003)
43. J Chen, M Gerla, Y Lee, M Sanadidi, TCP with delayed ack for wireless networks. Ad Hoc Netw. **6**(7), 1098–1116 (2008). doi:10.1016/j.adhoc.2007.10.004
44. AM Al-Jubari, M Othman, A new delayed ACK strategy for TCP in multi-hop wireless networks. in *Proceeding of International Information Technology Symposium (ITSim)*. **2**, 946–951 (2010)
45. FR Armaghani, SS Jamuar, S Khatun, MFA Rasid, Performance analysis of TCP with delayed acknowledgments in multi-hop ad-hoc networks. Wirel. Personal Commun. **56**, 791–811 (2009)
46. Z Fu, H Luo, P Zerfos, S Lu, L Zhang, M Gerla, The impact of multihop wireless channel on TCP performance. IEEE Trans Mobile Comput. **4**(2), 209–221 (2005)
47. L Ding, X Wang, Y Xu, W Zhang, Improve throughput of TCP-Vegas in multihop ad hoc networks?. Comput Commun. **31**(10), 2581–2588 (2008). doi:10.1016/j.comcom.2008.03.026
48. K Nahm, A Helmy, C Jay Kuo, TCP over multihop 802.11 networks: issues and performance enhancement. in *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing* 277–287 (2005)
49. X Wang, Y Han, Y Xu, APS-FeW: Improving TCP throughput over multihop ad hoc networks. Comput Commun. **32**(1), 19–24 (2009). doi:10.1016/j.comcom.2008.08.024
50. S Floyd, V Jacobson, Random early detection gateways for congestion avoidance. IEEE/ACM Trans Netw. **1**(4), 397–413 (1993). doi:10.1109/90.251892

doi:10.1186/1687-1499-2011-198
Cite this article as: Al-Jubari *et al.*: **TCP performance in multi-hop wireless ad hoc networks: challenges and solution.** *EURASIP Journal on Wireless Communications and Networking* 2011 **2011**:198.