


A Feedback Aware Reliable Transport Protocol with Improved Window Increment Mechanism for Inter Vehicular Wireless Network

M. Joseph Auxilius Jude¹  · V. C. Diniesh¹ · M. Shivaranjani¹ · R. Shanju¹

Published online: 21 August 2017
© Springer Science+Business Media, LLC 2017

Abstract In recent years, inter vehicular networking technologies are gaining momentum due to its merger with wireless internet through infrastructure gateways. A large quantity of internet data traffic heavily relies on transmission control protocol (TCP) due to reliable connection oriented services. TCP with sluggish window growth and weakly shaped congestion control mechanism fails under multi-hop inter vehicular conditions due to disjoint links and wireless channel errors. This paper introduces inter vehicular access network (IVAN) TCP, designed to suppress the limitations of congestion control and window increment algorithm in the multi-hop inter vehicular wireless environment. The IVAN's flexible window increment algorithm newer growth pattern achieves a faster increase in the transmission rate. Furthermore, IVAN's congestion control algorithm reduces the transmission rate based on sender's window utility and relay router's explicit congestion notification. The simulation results confirm the significant rise in IVAN's throughput performance, substantial reduction in the queue packet drop and packet latency against standard TCP variants under inter vehicular environment.

Keywords Vehicular ad hoc network (VANET) · Transmission control protocol (TCP) · Slow start (SS) · Flexible window increment (FWI) · Inter vehicular access network (IVAN)

✉ M. Joseph Auxilius Jude
jude2193@gmail.com

V. C. Diniesh
vcdiniesh@gmail.com

M. Shivaranjani
ranjani093@gmail.com

R. Shanju
shanjusri@gmail.com

¹ Self Organised Networking Group (SONG), Department of Electronics and Communication Engineering, Kongu Engineering College, Perundurai, Tamilnadu 638060, India

1 Introduction

Vehicular ad-hoc network (VANET) [1–4] developed as a part of Intelligent transportation system (ITS) and gaining popularity due to the wider range of applications such as traffic sharing, road navigation, automatic toll payment, Geo-location services and internet access through the roadside unit (RSU). The VANET operates under vehicle-to-vehicle (V2V), vehicle-to-Infrastructure (V2I) mode, and hybrid mode. The V2V mode uses the onboard unit (OBU) to communicate between vehicles. The V2I mode relies on OBU and RSU for data sharing between vehicle and infrastructure. The hybrid mode supports both V2I and V2V communication modes (Fig. 1).

The Institute of Electrical and Electronics Engineers (IEEE) exclusively allocated two standards, IEEE 802.11p [5] for physical layer interaction between vehicles commonly known as dedicated short range communications (DSRC) and IEEE 1609.X [6–8], wireless access in vehicular environments (WAVE) for the higher layer interaction. The IEEE 802.11p uses seven non overlapping channels with the spectrum of 75 MHz and channel width of 10 MHz, one channel dedicated to carrying critical traffic related control information and the rest of them for non critical service information.

Recent years witnessed a sharp increase in the usage of mobile internet through VANET's RSU gateways, enhances transportation and infotainment capabilities. The internet based VANET [9–11] strengthens driver's situational awareness with improved global connectivity for both safety and non safety related applications. The VANET merger with internet introduces the performance based challenge for TCP [12–15] under vehicular access environment. TCP [16, 17], a prominent and widely used transport protocol that carries a large chunk of internet data traffic plays a crucial role in determining the internet's performance. The TCP's congestion control algorithm infers a packet loss through missing acknowledgments (ACKs) and flatly reduces the dispatching rate with the assumption of network congestion in the relay routers.

1.1 Slower Window Increment Rate

TCP doubles the congestion window ($cwnd$) for each successful ACK in the slow start (SS) phase. In the congestion avoidance (CA) phase, $cwnd$ increments one segment for each roundtrip time (RTT). The additive increase and multiplicative decrease $cwnd$ growth pattern of the SS and CA phase require additional RTTs to finish the flows that cripple throughput performance i.e. slow $cwnd$ growth rate affects TCP throughput performance.

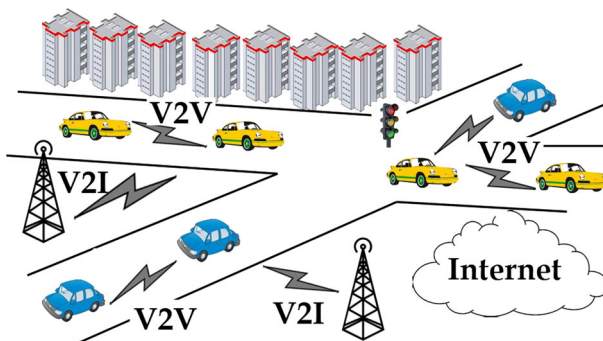


Fig. 1 VANET with mobile internet architecture model

$$lcwnd_{i+1} = \begin{cases} cwnd_i + 1, & SS \\ cwnd_i + \frac{1}{cwnd_i}, & CA \end{cases} \quad (1)$$

1.2 Flat Dispatching Rate Reduction

During the congestion control process, TCP flatly reduces the transmission rate by half irrespective of their utilization levels. The flat transmission rate reduction harshly degrades the least utilized TCP flows throughput performance. The new slow start threshold (*ssthresh*) value i.e. transmission rate reduction given in the Eq. (2)

$$ssthresh = \frac{cwnd_i}{2} \quad (2)$$

The weakly structured TCP congestion control algorithm collapses under wireless environment as it fails to recognize and isolate congested wireless link losses from the wireless error losses, results in needless transmission rate reduction. Past studies and analysis [12–15] reveal TCP's inability to suppress false window reduction results in throughput degradation and poor link utilization. However, explicit congestion notification (ECN) [18] based packet marking algorithms [19–28] proposed to enhance TCP's performance in the multi-hop wireless environment, fail to address slow *cwnd* increment rate, and flat transmission rate reduction. The IVAN's FWI algorithm and window utility based rate reduction proposed in this paper suppress TCP's limitations in vehicular networks.

2 Background and Related Works

This section primarily reviews the recent progress in the ECN based packet marking algorithms developed for multi-hop wireless networks. TCP-optimal queue selection (TCP-OQS) [19] approach proposed to enhance throughput largely depends on the relay nodes dual queuing algorithm that separately buffers data and ACK packets. The relay nodes initiate ECN when the packet arrival rate exceeds the queue threshold. In the sender side, TCP-OQS supports additive increase window growth algorithm and rate reduction based on relay router's queue length information.

The joint window based congestion control approach and rate-based distributed scheduling algorithm proposed in the paper [20] heavily relies on virtual rate estimation. The window increment algorithm follows additive increase growth pattern and initiates transmission rate reduction from the missing ACKs. The scheduling algorithm estimates virtual rate from the packet arrival rate and link capacity. The window based flow control approach remain defenseless against wireless error losses. The non-congestion events (NCE) [21] algorithm, a modified form of ECN/queue length based NewReno [29] algorithm overcomes the false rate reduction due to the non-congested random loss in wireless networks. The ECN helps the sender to avoid spurious transmission rate reduction and faster packet retransmission during random losses.

Non-congested retransmission timeouts (NRT) [22] TCP identifies wireless losses using NRT-detection, NRT-differentiation, and NRT-reaction. During timeouts, the sender confirms ECN marked ACK packets as the indication of network congestion and trims

down the dispatching rate. Ironically, the sender considers unmarked ACK packets as spurious timeout and re-transmits the old packet without reducing the transmission rate.

Hosam El-Ocla proposed congestion control enhancement for random loss (CERL) [23], a sender side modified version of NewReno. The CERL algorithm identifies the bottleneck link loss from the RTT latency measurements that includes queuing latency, round trip propagation time latency, and service latency. The packet latency based CERL congestion control algorithm prevents needless rate reduction for wireless transmission losses.

The explicit link failure notification (ELN) TCP [24] identifies non-congested wireless losses from the wireless relay router's re-transmission start notification message. The ECN based flow control mechanism avoids spurious timeouts, regulate packet reordering and limits *cwnd* shrinking associated with multiple packet drops. The feedback assisted congestion control algorithm [25] uses queue length information and the advertised window (*awnd*) value to adjust the dispatching rate. The sender uses RTT latency and queue length information to fix the dispatching rate that achieves fairness among different TCP flows.

The combined TCP bandwidth allocation (TBA) algorithm and wireless explicit congestion notification (WECN) mechanism [26] proposed to improve fairness over the heterogeneous wireless networks and estimate the optimal window size. The TBA/WECN approach divided into three functional parts: TBA calculation, ACK identification and writing ACK option. The ACK identification recognizes and detects the ECN bit set of the downstream wireless nodes. The writing ACK option writes the suitable window size in the advertised window of each ACK packet. The sender initiate flow control based on relay nodes WECN marked echo packets.

The TCP-Jersey [27] consists of two key components, the available bandwidth estimation (ABE) [30] algorithm, and the congestion warning (CW) router configuration. At the sender side, ABE estimates the wireless links available bandwidth and helps the sender to re-adjust the transmission rate. In the case of network congestion, the relay routers alert the end stations by marking CW field in the packets. The relay routers packet marking helps the sender to discriminate the packet losses caused by network congestion from wireless link errors. TCP New Jersey [28] minor modification of Jersey, uses timestamp option along with the ABE algorithm.

In addition to ECN based congestion control approach, several RTT delay based congestion control algorithms [31–35] proposed for multi-hop wireless network. In the delay based approach, TCP's window updates solely rely on RTT measurement i.e. observing TCP acknowledgments inter-arrival times. In multi-hop wireless networks, delay based congestion control remain vulnerable to spurious timeout that arises due to relay nodes re-routing during link failure.

In summary, TCP congestion control algorithm blended with packet marking mechanism focused on wireless error packet loss detection and isolation. The ECN based packet marking algorithm discussed in the literature fails to address TCP's inherent limitations such as slower window increment rate and the flat dispatching rate reduction during congestion control.

3 TCP IVAN

The IVAN customized for inter vehicular wireless access environment requires four major modifications.

Modification 1: The sender side *cwnd* growth algorithm i.e. FWI algorithm.

Modification 2: The relay node’s single queue dual marking (SQDM) algorithm.

Modification 3: The receiver’s congestion echo state variables.

Modification 4: The congestion control algorithm based on the window utility and relay router’s explicit congestion notification.

3.1 Flexible Window Increment (FWI) Algorithm

The sender side FWI algorithm divides the entire *cwnd* growth into five thresholds based on the receiver’s advertised window capacity. The sender uses the thresholds to monitor TCP flows growth level and initiates rate reduction based on the *cwnd* growth. The threshold γ_1 represents the initial window (*int_wnd*), γ_5 the maximum window (*max_wnd*), γ_3 centroid window, γ_2 lower midpoint between γ_1 and γ_3 , and γ_4 the upper midpoint between γ_3 and γ_5 . The threshold above γ_3 referred as optimum window utility and below as a lower window utility. In the optimum window utility, the *cwnd* growth rate exceeds 50% of the receiver advertised capacity and vice versa in the lower utility region. For each successful ACK, the FWI updates the transmission rate by computing current *cwnd*, γ_5 , and increment parameter (*inc_p*). The *inc_p* determines the *cwnd* growth and takes the value 0.125, higher *inc_p* value results in bottleneck link congestion and smaller *inc_p* value leads to slower window increment rate. The SS phase *cwnd* update function and the window threshold levels computed using the following equations.

$$cwnd_{i+1} = cwnd_i + (\gamma_5 - cwnd_i) * inc_p; \quad 0 < cwnd < \gamma_5 \tag{3}$$

$$\gamma_5 = max_wnd \tag{4}$$

$$\gamma_1 = int_wnd \tag{5}$$

$$\gamma_3 = \frac{\gamma_1 + \gamma_5}{2} \quad (\text{centroid}) \tag{6}$$

$$\gamma_2 = \frac{\gamma_1 + \gamma_3}{2} \quad (\text{lower midpoint}) \tag{7}$$

$$\gamma_4 = \frac{\gamma_3 + \gamma_5}{2} \quad (\text{upper midpoint}) \tag{8}$$

When the *cwnd* reaches the maximum receiver advertised capacity (γ_5), FWI ends the SS phase and moves to CA phase. In the maximum window condition i.e. $cwnd = \gamma_5$, the sender compares current RTT against the sum RTT i.e. the average of last three RTTs and sets the CA phase new *ssthresh*. The *ssthresh* and *cwnd* updated using the following equation.

$$ssthresh = \begin{cases} \gamma_5 * \frac{3}{4}; & RTT < Sum_RTT \\ \gamma_5 * \frac{1}{2}; & RTT \geq Sum_RTT \end{cases} \tag{9}$$

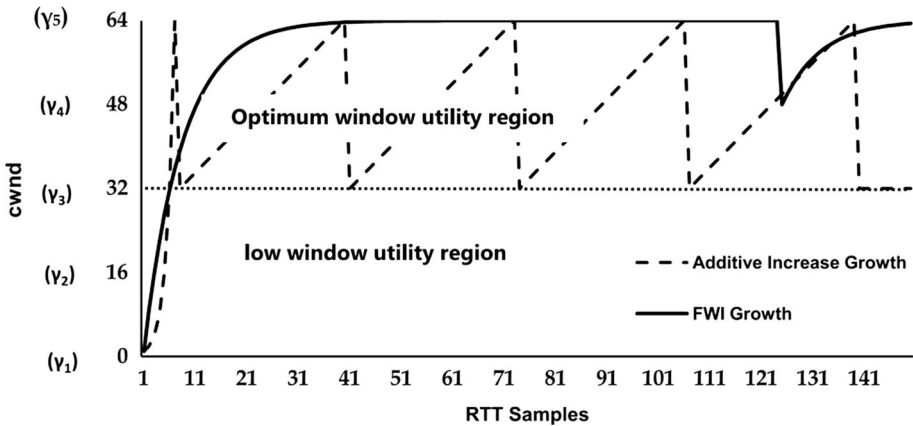


Fig. 2 Exponential versus flexible Window Growth

$$cwnd_i = ssthresh \tag{10}$$

$$cwnd_{i+1} = cwnd_i + \frac{(\gamma_5 - cwnd_i) * inc_p}{2} \tag{11}$$

The FWI *cwnd* update function achieves the faster growth rate and remains in the optimum window utility region for longer RTTs i.e. *cwnd* quickly reaches 50% receiver’s advertised capacity. From the Eqs. (1) to (12), theoretical FWI and additive increase growth computed with no loss condition and shown in the Fig. 2. For the *cwnd* computations, γ_1 set as 1 and γ_5 as 64 for the maximum duration of 150 RTTs. The FWI algorithm remains in the optimum window utility region for longer 118 RTTs and achieves better improvement in the throughput rate. On the other hand, the SS phase of additive increase window pattern ends within 7 RTTs and shifts to CA phase that substantially slows down the growth rate.

3.2 Single Queue Dual Marking (SQDM) Mechanism

The IVAN congestion control algorithm heavily depends on relay router’s packet marking information. The idea behind SQDM carved from the NCE’s ECN based queue mechanism [21] with minor modifications in the threshold settings and congestion notification, i.e. two level notifications. The existing NCE’s queuing algorithm with smaller queue threshold value falsely initiates spurious congestion alert during re-routing, results in needless transmission rate reduction. The SQDM approach maintains two threshold levels (Q_{Th1} and Q_{Th2}) to notify the sender about the relay nodes queue buildup. The Q_{Th1} fixed as 50% of average queue size, provides an adequate buffer capacity to store the packets during re-routing that occurs due to link breakage. The Q_{Th2} marking level chosen as 75% of the total average queue size (Q_{Total}), notifies the sender about the critical queue buildup in the relay routers (Fig. 3).

Algorithm 1 Pseudocode for SQDM

```

Initialization
 $Q_{Th1} = 1/2 * Q_{Total}$ 
 $Q_{Th2} = 3/4 * Q_{Total}$ 
 $Q_{Avg\_cur} = Q_{Avg\_prev} = 0$ 
for every data packet
    if ( $Q_{Avg\_cur} > Q_{Avg\_prev}$ )
        if  $Q_{Th1} < Q_{Avg\_cur} < Q_{Th2}$ 
            | mark CE flag
        end
        if  $Q_{Avg\_cur} > Q_{Th2}$ 
            | mark SCE flag
        end
    end
end
    
```

The SQDM interprets the queue accumulation above the Q_{Th1} and Q_{Th2} as a symptom of incipient and severe congestion, initiates first and second level congestion notification to the sender. The SQDM estimates queue accumulation from the current average queue size (Q_{Avg_cur}) [10] as

$$Q_{Avg(k+1)} = (1 - Q_{wt}) * Q_{Avg(k)} + Q_{wt} * Q_{cur(k)} \tag{12}$$

where Q_{cur} is the current queue size and Q_{wt} be the queue weight, takes the value 0.2 [27]. For every packet arrival, SQDM checks the status of the Q_{Avg_cur} , if the Q_{Avg_cur} exceeds Q_{Th1} , relay routers mark the incoming packets with the congestion experienced (CE) field in the internet protocol (IP) header [36] that alerts the sender about incipient network congestion. When the Q_{Avg_cur} exceeds Q_{Th2} , the relay router performs the second level marking by initiating a newly added severe congestion experience (SCE) flag in the IP packet header. The relay router dual level marking improves sender alertness towards network congestion. In short, the sender will have an update on appropriate congestion level and reduces the transmission rate accordingly. Furthermore, the SQDM algorithm with improved marking threshold overcomes the spurious congestion notification impact.

3.3 Receiver Echo Notification

The receiver enables explicit congestion notification echo (ECE)/severe congestion notification echo (SCNE) flag in the TCP header based on relay router’s CE/SCE marking. On

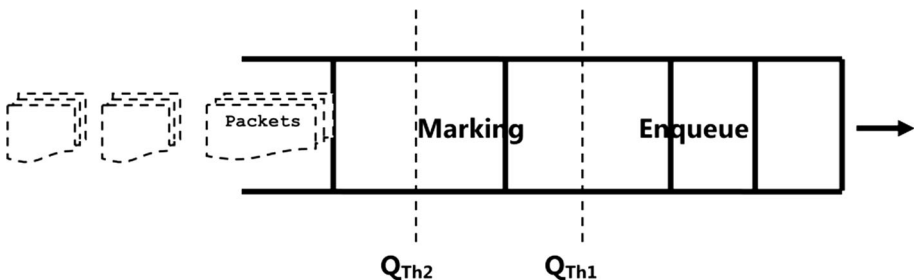


Fig. 3 Queue model: single queue dual marking (SQDM)

receiving the ECE/SCNE echo, the sender initiates congestion control process and quickly acknowledges the receiver about transmission rate reduction. The sender uses congestion window reduced (CWR) TCP flag for ECE/SCNE echo acknowledgment. The following limitations observed in the existing receiver echo notification that remain unsuitable for dual level congestion alert.

- i. On receiving the CWR response flag, the receiver stops enabling future congestion echo flags in the TCP header and remains ineffective against the relay nodes small sized buffer [37] frequent queue variations.
- ii. After receiving the first level CWR flag, the receiver fails to initiate second level notification i.e. SCNE notification.

To overcome the existing receiver echo notification limitations, IVAN uses two state variables approach as shown in the Fig. 4.

The receiver maintains packet marking-1 (PM1) and packet marking-2 (PM2) variables that precisely echoes the relay router’s congestion notification to the sender. For non-congested condition, the receiver state variables remain inactive (‘PM1 = PM2 = 0’). For first level CE notification, receiver enables ECE flag in the ACK packet to notify the incipient congestion. On receiving the first level CWR flag, the receiver stops first level echo and activates PM1 (‘PM1 = 1 and PM2 = 0’) i.e. no further marking of level 1 notification. For second level SCE packet marking, receiver enables SCNE flag to notify the severe congestion. On receiving the second level CWR response flag, receiver activates both the state variables (‘PM1 = PM2 = 1’). The state variables PM1 and PM2 reset to zero for no marking condition. The receiver congestion echoes based on two state variables swiftly react to queue buildup and supports two level congestion notification (Table 1).

3.4 Sender Congestion Control Algorithm

The proposed IVAN congestion control algorithm responds solely to receiver ECE/SCNE notification levels. For the first level ECE echo flag, IVAN reduces the transmission rate based on *cwnd* growth level. The *ssthresh* and *cwnd* of TCP flow above the γ_3 threshold level updated as

$$ssthresh = \frac{cwnd_i + \gamma_2}{2}; \quad cwnd \geq \gamma_3 \tag{13}$$

$$cwnd_{i+1} = cwnd_i + (\gamma_5 - cwnd_i) * dec_p1; \quad ssthresh < cwnd < \gamma_5 \tag{14}$$

where *dec_p1* is the first order decrement parameter and takes the value $\frac{1}{16}$

However, the *ssthresh* of TCP flow below the γ_3 threshold level updated as

$$ssthresh = \frac{cwnd_i + (0.25 * \gamma_2)}{2}; \quad cwnd < \gamma_3 \tag{15}$$

The *cwnd* updated using the Eq. (14)

While on receiving the second level SCNE marking, the sender initiates slow start and sets the flat *ssthresh* regardless of flow utility levels. In the congestion control process, the *cwnd* follows an additive increase pattern until the *ssthresh* limit and a newer *cwnd* increment pattern beyond the *ssthresh*. The level 2 notification *ssthresh* and *cwnd* updated as

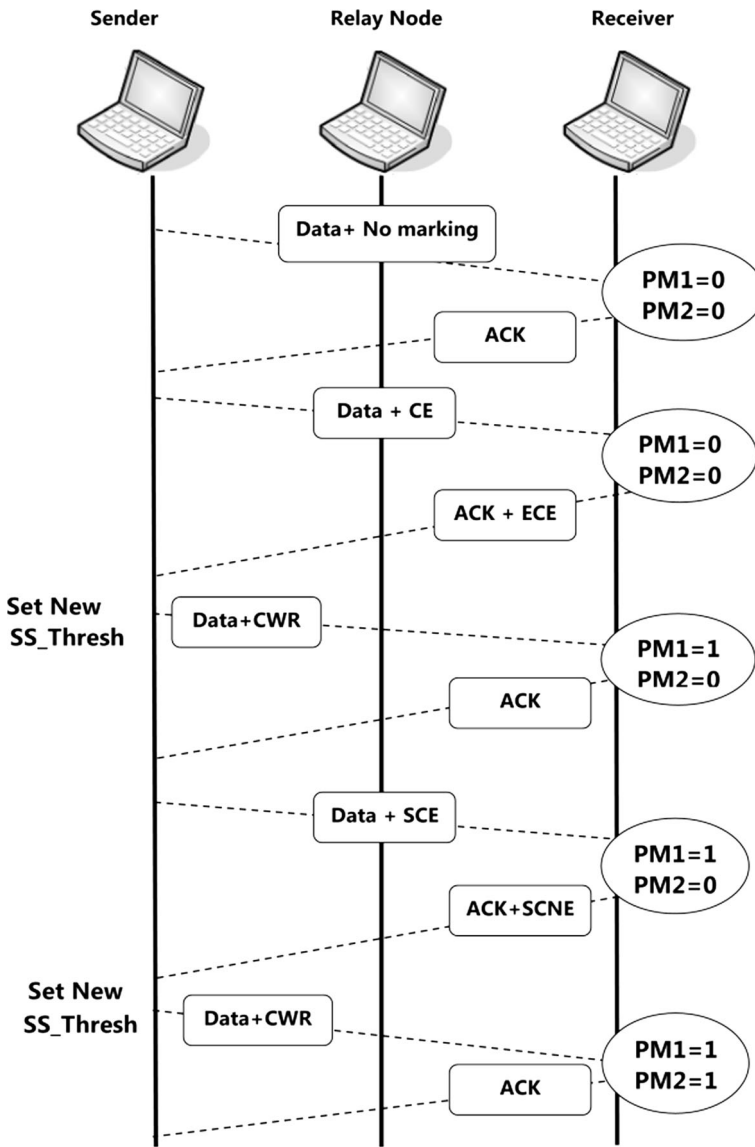


Fig. 4 TCP IVAN flow diagram

Table 1 Receiver state variable update

Relay node notification	Receiver echo flag	Sender CWR flag	State variable update
No marking	–	–	PM1 = 0 PM2 = 0
CE	ECE	CWR	PM1 = 1 PM2 = 0
SCE	SCNE	CWR	PM1 = 1 PM2 = 1

$$ssthresh = \frac{cwnd_i + \gamma_2}{4} \tag{16}$$

$$cwnd_{i+1} = \begin{cases} cwnd_i + 1; & cwnd < ssthresh \\ cwnd_i + (\gamma_5 - cwnd_i) * dec_p2; & cwnd \geq ssthresh \end{cases} \tag{17}$$

where dec_p2 is the second order decrement parameter and takes the value $\frac{1}{32}$

In short, IVAN congestion control algorithm uses three different scales (Eq. (13), (15) and (16)) to reduce the transmission rate based on the flow utilization level. Furthermore, the $cwnd$ growth in the congestion control phase achieves a substantial improvement in recovery rate with lesser buffer packet drops over the standard TCP’s exponential growth mechanism.

Algorithm 2 Pseudocode for IVAN operation

Flexible Window Increment

```

For every RTT
    if (cwnd <  $\gamma_3$ ) (SS phase)
        |
        |   cwndi+1 = cwndi + ( $\gamma_5$  - cwndi) * inc_p
        |
    else (cwnd  $\geq$   $\gamma_3$ ) (CA phase)
        |
        |   set ssthresh =  $\gamma_5 * \frac{3}{4}$  ; RTT < Sum_RTT
        |
        |    $\gamma_5 * \frac{1}{2}$  ; RTT  $\geq$  Sum_RTT
        |
        |   cwndi+1 = cwndi + ( $\gamma_5$  - cwndi) * inc_p
    end
end
    
```

end

Congestion Control

```

For every RTT
    if SCNE received
        |
        |   set ssthresh =  $\frac{cwnd_i + \gamma_2}{4}$ 
        |
        |   ssthresh = cwndi
        |   cwndi+1 = cwndi + 1, for each ACK ; cwnd < ssthresh
        |
        |    $cwnd_i + (\gamma_5 - cwnd_i) * dec\_p2$ ; cwnd  $\geq$  ssthresh
    else if ECE received
        |
        |   if cwndi >  $\gamma_3$ 
        |   |
        |   |   set ssthresh =  $\frac{cwnd_i + \gamma_2}{2}$ 
        |   |
        |   |   ssthresh = cwndi
        |   |   cwndi+1 = cwndi + ( $\gamma_5 - cwnd_i$ ) * dec_p1 *
        |   |
        |   |   else
        |   |   |   set ssthresh =  $\frac{cwnd + (0.25 * \gamma_3)}{2}$ 
        |   |   |
        |   |   |   ssthresh = cwndi
        |   |   |   cwndi+1 = cwndi + ( $\gamma_5 - cwnd_i$ ) * dec_p1 *
        |   |   |
        |   |   end
        |   |
        |   end
    end
    if (3DUPACK) / timeout
        |
        |   retransmit the packet
    end
end
end
    
```

Table 2 Simulation parameters

General parameters	
No. of nodes (vehicles)	150
Propagation model	Nakagami
Area	1150 * 1150 meters
Simulation time	900 s
Antenna height and type	1.5 m and Omni Direction
Frequency	5.29 GHz
MAC	802.11 p
Data rate	27 Mb/s
Mobility	10 m/s for city, 15 m/s for sub-urban and 20 m/s for highway
TCP	IVAN, NCE and Westwood
Packet size	512
TCP flows	2,4,6, and 8

4 Performance Analysis

In this section, an extensive experimental analysis of IVAN was carried out against two standard TCP variants (NCE and Westwood) under multi-hop vehicular environment. TCP Westwood, NCE, and IVAN use the different queuing approach in the relay nodes, i.e. Westwood uses FIFO, NCE uses ECN based active queue, and IVAN uses SQDM queuing algorithm. The vehicular mobility scenarios created using VanetMobisim [38] with roads, traffic lights, car movement, edges, turns, etc., The network simulator (NS-2) [39] used for higher layer protocol interaction (routing, transport protocol, and FTP traffic model). Two sets of experiments carried out to measure the congestion control algorithm and queuing performance under re-routing and varying wireless traffic conditions. To evaluate the performance of IVAN, NCE and Westwood, the four metrics namely, throughput, end-to-end packet latency, packet loss rate and packet drop chosen as per the guidelines of RFC 5166 [40]. Table 2 gives a more detailed description of simulation parameters used in the creation of vehicular experimental setup.

4.1 Mobility Versus Metrics

The primary analysis measures the standard TCP's congestion control algorithm performance against the IVAN under high mobility condition. The number of vehicular nodes chosen as 100 for urban, sun-urban and highway scenarios. The metrics evaluated by varying the vehicle speed, i.e. 10, 15 and 20 m/s for urban, sun-urban and highway scenarios. The mobility scenario also measures the queuing algorithm performance during re-routing associated with link failure. The proposed IVAN uses SQDM queue, NCE enabled with ECN and Westwood uses FIFO queuing mechanism.

The graph in the Fig. 5 clearly illustrates the severe throughput degradation of Westwood and NCE performance under high mobility conditions. The FIFO based Westwood TCP misinterprets link failure packet drop as an early indication of congestion and cut down the dispatching rate. Furthermore, the existing Westwood and NCE slow window growth rate result in higher packet latency and slow throughput rate. The ECN based NCE

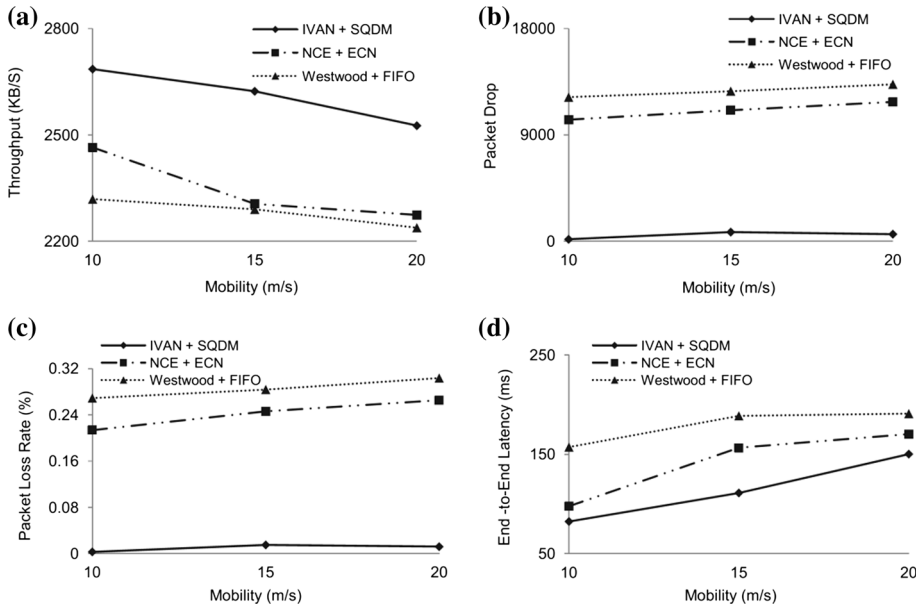


Fig. 5 Mobility versus Metrics. **a** Throughput. **b** Packet drop. **c** Packet loss rate. **d** End-to-end latency

congestion control algorithm *cwnd* growth suffers due to spurious congestion alert limitations that severely deteriorates throughput performance.

On the other hand, IVAN uses higher QTh1 SQDM queues (50% reserve for storage) have an adequate capacity to hold the packets during re-routing that avoids spurious congestion notification. IVAN congestion control algorithm improves throughput by a significant margin (12.6 and 10.2% against Westwood and NCE), reduction in buffer overflows (95.4 and 94.8% against Westwood and NCE), loss rate (96 and 95.3% against Westwood and NCE) and end-to-end latency (21 and 12% against Westwood and NCE).

4.2 Number of Active TCP Traffic Flow Versus Metrics

The second analysis measures the TCP's traffic handling capacity and queuing algorithm performance in the inter-vehicular environment. TCP traffic varies from a minimum of 2 connections to a maximum of 8 connections with the average velocity of 10 m/s (Fig. 6).

In high traffic conditions, the packet arrival rate exceeds relay routers processing rate and quickly overwhelms the buffer capacity. The IVAN's SQDM queuing algorithm initiate level 1 and level 2 congestion notification that stabilizes the flow rate in the relay nodes. The IVAN's *cwnd* rate reduction based on the flow utilization level improves recovery mechanism. In contrast, NCE flatly reduces the dispatching rate by half that cripples lower window utility flows throughput performance. The effectiveness of IVAN's improved recovery mechanism and dual level SQDM queue mechanism reflected in the overall throughput performance improvement (12 and 10.6% against Westwood and NCE). Furthermore, IVAN achieves substantial reduction in buffer packet drop (97.4 and 97.3% against Westwood and NCE), loss rate (97.8 and 97% against Westwood and NCE) and end-to-end packet latency (20 and 15.5% against Westwood and NCE).

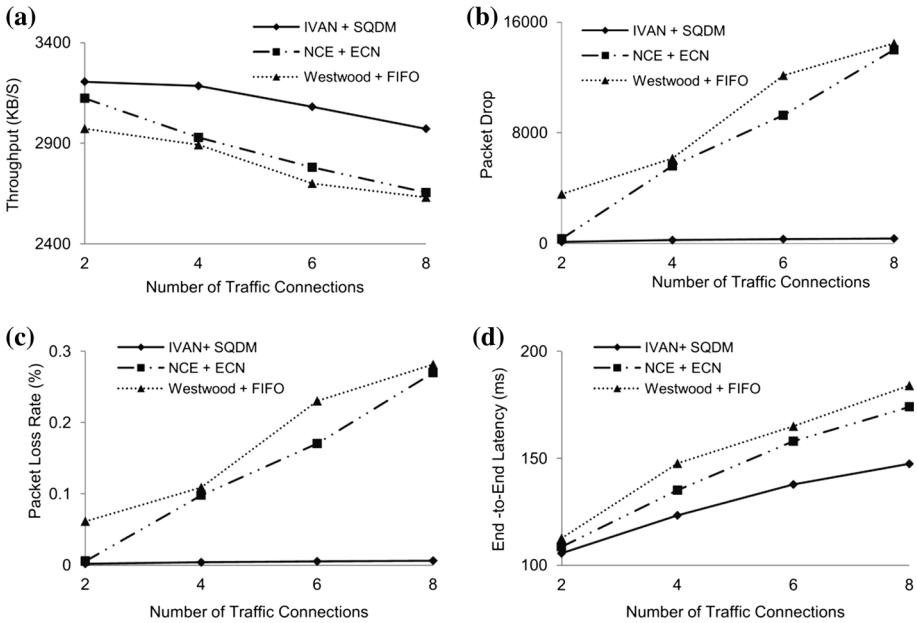


Fig. 6 Traffic Connections versus Metrics. **a** Throughput. **b** Packet drop. **c** Packet loss rate. **d** End-to-end latency

5 Conclusion

The IVAN TCP proposed in this paper overcomes flat window reduction and slower window increment rate limitations in the multi-hop inter vehicular environment. The IVAN uses the relay routers two level congestion notification to identify the packet drop of wireless channel errors from the congested link loss. The ECE/SCNE flag substantially improves sender’s decision to initiate congestion control during packet drops that eliminate the false window reduction. The FWI algorithm uses a unique window increment pattern that accelerates the transmission rate in the SS and CA phase. The IVAN fixes new ssthresh based on TCP flows utility level that suppresses flat window reduction and improves throughput performance. The inter vehicular simulation outcome highlights the proposed IVAN TCP achieves improvement in throughput and substantial reduction in loss rate, packet drops, and end-to-end packet latency reduction against Westwood and NCE.

References

1. Zeadally, S., Hunt, R., Chen, Y., et al. (2012). Vehicular ad hoc networks (VANETS): Status, results, and challenges. *Springer Journal of Telecommunication Systems*. doi:10.1007/s11235-010-9400-5.
2. Al-Sultan, S., Al-Doori, M. M., Al-Bayatti, A. H., & Zedan, H. (2014). A comprehensive survey on vehicular ad hoc network. *Elsevier Network and Computer Applications*. doi:10.1016/j.jnca.2013.02.036.
3. Karagiannis, G., Altintas, O., Ekici, E., Heijenk, G., Jarupan, B., Lin, K., et al. (2011). Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions. *IEEE Communications Surveys & Tutorials*. doi:10.1109/SURV.2011.061411.00019.

4. Cunha, F., Villas, L., Boukerche, A., et al. (2016). Data communication in VANETs: Protocols, applications and challenges. *Elsevier Ad Hoc Networks*. doi:[10.1016/j.adhoc.2016.02.017](https://doi.org/10.1016/j.adhoc.2016.02.017).
5. IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. (2007). Std:802.11-2007.
6. IEEE 1609.0/D0.7. Draft Standard for Wireless Access in Vehicular Environments (WAVE)—Architecture. (2009).
7. IEEE 1609.4/D6.0. (2010). Draft Standard for Wireless Access in Vehicular Environments (WAVE)—Multi-channel Operation in IEEE Vehicular Technology Society.
8. IEEE 1609.3/D5.0. (2010). Trial-Use Standard for Wireless Access in Vehicular Environments (WAVE)—Networking Services. IEEE Vehicular Technology Society.
9. Gerla, M., & Kleinrock, L. (2011). Vehicular networks and the future of the mobile internet. *Elsevier Computer Networks*. doi:[10.1016/j.comnet.2010.10.015](https://doi.org/10.1016/j.comnet.2010.10.015).
10. Campolo, C., Molinaro, A., & Scopigno, R. (2015). From today's VANETs to tomorrow's planning and the bets for the day after. *Elsevier Vehicular Communications*. doi:[10.1016/j.vehcom.2015.06.002](https://doi.org/10.1016/j.vehcom.2015.06.002).
11. Baid, A., Mukherjee, S., & Vu, T. (2013). Enabling vehicular networking in the mobility first future internet architecture. *Proceedings of IEEE WoWMoM*. doi:[10.1109/WoWMoM.2013.6583417](https://doi.org/10.1109/WoWMoM.2013.6583417).
12. Jude, J. A., & Ganesan, R. (2015). A comprehensive experimental analysis of standard TCP variants in vehicular environment. *Proceedings of IEEE ICCCT*. doi:[10.1109/ICCCT2.2015.7292774](https://doi.org/10.1109/ICCCT2.2015.7292774).
13. Zhang, L., Lakas, A., El-Sayed, H., & Barka, E. (2013). Mobility analysis in vehicular ad-hoc network (VANET). *Elsevier Network and Computer Applications*. doi:[10.1016/j.jnca.2012.12.008](https://doi.org/10.1016/j.jnca.2012.12.008).
14. Mast, N., & Owens, T. J. (2011). A survey of performance enhancement of transmission control protocol (TCP) in wireless ad hoc networks. *EURASIP Wireless Communications and Networking*. doi:[10.1186/1687-1499-2011-96](https://doi.org/10.1186/1687-1499-2011-96).
15. Al-Jubari, A. M., Othman, M., Mohd Ali, B., et al. (2011). TCP performance in multi-hop wireless ad hoc networks: Challenges and solution. *EURASIP Wireless Communications and Networking*. doi:[10.1186/1687-1499-2011-198](https://doi.org/10.1186/1687-1499-2011-198).
16. Postel, J. (1981). *Transmission Control Protocol*. RFC 793, IETF Network Working Group.
17. Allman, M., Paxson, V., & Stevens, W. (1999). *TCP Congestion Control*. RFC 2581, IETF Network Working Group.
18. Ramakrishnan, K., Floyd, S., & Black, D. (2001). *The Addition of Explicit Congestion Notification (ECN) to IP*. RFC 3168, IETF Network Working Group.
19. Sathya Priya, S., & Murugan, K. (2015). Enhancing TCP fairness in wireless networks using dual queue approach with optimal queue selection. *Springer Wireless Personal Communication*. doi:[10.1007/s11277-015-2455-6](https://doi.org/10.1007/s11277-015-2455-6).
20. Huang, P.-K., Lin, X., & Wang, C.-C. (2013). a low-complexity congestion control and scheduling algorithm for multihop wireless networks with order-optimal per-flow delay. *IEEE/ACM Transactions on Networking*. doi:[10.1109/TNET.2012.2213343](https://doi.org/10.1109/TNET.2012.2213343).
21. Sreekumari, P., & Chung, S.-H. (2011). TCP NCE: A unified solution for non-congestion events to improve the performance of TCP over wireless networks. *EURASIP Wireless Communications and Networking*. doi:[10.1186/1687-1499-2011-23](https://doi.org/10.1186/1687-1499-2011-23).
22. Sreekumari, P., & Lee, M. (2013). TCP NRT: A new TCP algorithm for differentiating non-congestion retransmission timeouts over multi-hop wireless networks. *EURASIP Wireless Communications and Networking*. doi:[10.1186/1687-1499-2013-172](https://doi.org/10.1186/1687-1499-2013-172).
23. El-Ocla, H. (2010). TCP CERL: Congestion control enhancement over wireless Networks. *Springer Wireless Networks*. doi:[10.1007/s11276-008-0123-4](https://doi.org/10.1007/s11276-008-0123-4).
24. Yun, J.-H. (2009). Cross-layer explicit link status notification to improve TCP performance in wireless networks. *EURASIP Wireless Communications and Networking*. doi:[10.1155/2009/617818](https://doi.org/10.1155/2009/617818).
25. Byun, H. J., & Lim, J. T. (2005). Explicit window adaptation algorithm over TCP wireless networks. *IEE Proceedings—Communications*. doi:[10.1049/ip-com:20045091](https://doi.org/10.1049/ip-com:20045091).
26. Peng, F., & Leung, V. C. M. (2007). Enhancing fairness and throughput of TCP in heterogeneous wireless access networks. *Springer International Journal of Wireless Information Networks*. doi:[10.1109/PIMRC.2003.1264239](https://doi.org/10.1109/PIMRC.2003.1264239).
27. Xu, K., Tian, Y., & Ansari, N. (2004). TCP-jersey for wireless IP communication. *IEEE Transactions on Selected Areas in Communications*. doi:[10.1109/JSAC.2004.825989](https://doi.org/10.1109/JSAC.2004.825989).
28. Kai, Xu, Tian, Ye, & Ansari, Nirvan. (2005). Improving TCP performance in integrated wireless communications networks. *Elsevier Computer Networks*. doi:[10.1016/j.comnet.2004.07.006](https://doi.org/10.1016/j.comnet.2004.07.006).
29. Henderson, et al. (2012). *The NewReno Modification to TCP's Fast Recovery Algorithm*. RFC 6582, IETF Network Working Group.

30. Casetti, C., Gerla, M., Mascolo, S., Sanadidi, M. Y., & Wang, R. (2002). TCP westwood: end-to-end congestion control for wired/wireless networks. *Springer Wireless Networks*. doi:[10.1023/A:1016590112381](https://doi.org/10.1023/A:1016590112381).
31. Jiang, H., Luo, Y., Zhang, Q. Y., Yin, M. Y., & Wu, C. (2016). TCP-gvegas with prediction and adaptation in multi-hop ad hoc networks. *Journal of Wireless Networks*. doi:[10.1007/s11276-016-1242-y](https://doi.org/10.1007/s11276-016-1242-y).
32. Jamalia, S., Alipasandib, N., & Alipasandic, B. (2015). TCP pegas: A PSO-based improvement over TCP vegas. *Elsevier Applied Soft Computing*. doi:[10.1016/j.asoc.2015.03.048](https://doi.org/10.1016/j.asoc.2015.03.048).
33. Jung, S., Lee, J., Lee, G., Pyun, S.-Y., & Cho, D.-H. (2014). Novel fastest retransmission and rate control schemes for improving TCP performance in wireless ad hoc networks. *Journal of Wireless Personal Communications*. doi:[10.1007/s11277-013-1376-5](https://doi.org/10.1007/s11277-013-1376-5).
34. Badarla, V., & Siva Ram Murthy., C. (2011). Learning-TCP: A stochastic approach for efficient update in TCP congestion window in ad hoc wireless networks. *Journal of Parallel and Distributed Computing*. doi:[10.1016/j.jpdc.2010.12.012](https://doi.org/10.1016/j.jpdc.2010.12.012).
35. Utsumia, S., & Zabirc, S. M. S. (2014). A new high-performance TCP friendly congestion control over wireless networks. *Elsevier Journal of Network and Computer Applications*. doi:[10.1016/j.jnca.2014.02.003](https://doi.org/10.1016/j.jnca.2014.02.003).
36. Postel, J. (1981). *Internet Protocol*. RFC 791, IETF Network Working Group.
37. Showail, A., Jamshaid, K., & Shihada, B. (2016). Buffer sizing in wireless networks: Challenges, solutions, and opportunities. *IEEE Communications Magazine*. doi:[10.1109/MCOM.2016.7452277](https://doi.org/10.1109/MCOM.2016.7452277).
38. VanetMobsim:microscopic and macroscopic vehicular traffic simulator tool. <http://vanet.eurecom.fr>.
39. NS-2. Network Simulator (NS-2.34). <http://www.isis.edu/nsnam/ns>.
40. Floyd, S. (2008). *Metrics for the Evaluation of Congestion Control Mechanisms*. RFC 5166, IETF Network Working Group.



M. Joseph Auxilius Jude received his B.E in Electronics and Communication Engineering from St. Xavier's Catholic College of Engineering (SXCCE) and M.E with specialization in Digital Communication and Networking under from Arulmigu Kalasalingam College of Engineering (AKCE) Anna University, Chennai, India in 2005 and 2007 respectively. He is currently a Ph.D. Candidate in the Department of Information and Communication Engineering at Kongu Engineering College and pursuing research in the area of congestion control in self organized wireless networks. He is with Self Organized Networking Group (SONG) of the ECE department at Kongu Engineering College. He is closely associated with the development of advanced wireless networking lab and organised funded industry-aligned training programmes in alliance with Defence Research and Development Organisation (DRDO), Department of Science and Technology (DST) and Industry Institute Partnership Cell. His research interests include designing, testing and validation of scheduling algo-

rithm for wireless sensor networks, congestion control algorithms (TCP and AQM) for the mobile ad-hoc network, vehicular ad-hoc network and cognitive radio network. He is a member of IEEE, IETE and institution chair of IEEE computer society.



V. C. Diniesh received his B.E in Electronics and Communication Engineering from Arunai Engineering College (AEC) and M.E with specialization in Embedded and Real Time System from Coimbatore Institute of Technology (CIT) under Anna University, Chennai, India in 2009 and 2011 respectively. He is with Self Organized Networking Group (SONG) of the ECE department at Kongu Engineering College. He is closely connected with the routing protocol development and training for tactical wireless networks especially for disaster and battlefield management, organized funded programmes in collaboration with DST and Industry Institute Partnership. His research interests include designing, testing and validation of routing protocols for mobile ad-hoc network and wireless sensor network. He is a member of IEEE and IETE.



M. Shivaranjani completed her B.E in Electronics and Communication Engineering and Master Degree in Communication Systems, from Kongu Engineering College, an autonomous institution affiliated to Anna University, Chennai. She is with Self Organized Networking Group (SONG) of the ECE department at Kongu Engineering College. Her current research focuses on designing Transmission Control Protocol (TCP) for Wireless Multi-hop Networks especially Vehicular ad hoc network, Mobile ad hoc network and Cognitive Radio Network.



R. Shanju completed her Master Degree in Communication Systems, from Kongu Engineering College, an autonomous institution affiliated to Anna University, Chennai. She received her Bachelor degree in Electronics and Communication Engineering, from Angel College of Engineering and Technology affiliated to Anna University Chennai. She is with Self Organized Networking Group (SONG) of the ECE department at Kongu Engineering College. Her research interests are designing TCP for data center networks, ad-hoc networks and Cognitive Radio Network.