Proceedings of the International Conference on Intelligent Computing and Communication for Smart World (I2C2SW-2018)
14-15 December 2018

Improving TCP's Throughput and Fairness Stability in Vehicular Network

M. Joseph Auxilius Jude
Department of ECE
Kongu Engineering College
Perundurai, India
jude2193@gmail.com

S.Kuppuswami
Principal
Kongu Engineering College
Perundurai, India
skuppu@gmail.com

V. C. Diniesh
Department of ECE
Kongu Engineering College
Perundurai, India
vcdiniesh@gmail.com

***Abstract*—Throughput and fairness performance of transmission control protocol (TCP) worsens critically under vehicular environment due to its weakly bounded congestion threshold parameter, rate reduction approach and congestion recovery algorithm. In this paper, a new congestion control algorithm for vehicular networks, called VC-TCP (Vehicular customized -TCP) is proposed to overcome the limitations of the existing independent congestion control approaches. The VC-TCP implements a newer congestion threshold parameter, which regulates the spurious rate reduction due to an abrupt increase in RTT spike value. During the congestion control process, VC-TCP uses a new recovery algorithm and bandwidth utilization based rate reduction, which substantially improves the flow fairness of TCP traffic flows with different bandwidth utilization levels. The throughput, flow fairness, and packet latency performances of VC-TCP is evaluated against the standard Vegas congestion control approach under simulated vehicular scenarios. The simulation outcomes verify the performance improvement of the proposed VC-TCP approach against the standard TCP variants in the different facets of the vehicular environment.**

Keywords*—*Vehicular Ad-Hoc Network (VANET), Transmission Control Protocol (TCP), Intelligent Transportation Systems (ITS)

I. INTRODUCTION

Vehicular ad-hoc network (VANET) [1] is a distinct category of multi-hop ad-hoc networks established between vehicles with (or) without roadside infrastructure. VANETs equipped with radio communication devices are specifically designed to support vehicular safety applications which enhances passenger safety, driver assistance along with wireless internet capability. VANET remains an integral part of intelligent transportation systems (ITS) [2] which significantly enhances the transportation systems performances by reducing traffic congestion, road safety, idle waiting times in the tollgate, and fuel consumptions. VANET architecture basically operates under vehicle-to-vehicle (V2V) mode and vehicle-to-infrastructure (V2I) mode. Recent years witnessed a rapid proliferation of internet in vehicular networks due to the merger of inter-vehicular network with the readily available roadside hotspots. Fig. 1 shows the VANET architecture for V2I and V2V scenarios.

The substantial part of the algorithms and protocols implemented in the vehicular networks are derived from the proven technologies developed for cellular networks or 802.11 networks (wireless local area networks (WLANs). However, congestion control algorithm developed for single hop networks (cellular technology (or) WLAN) remains incompetent to support vehicular wireless internet as it encounters frequent link failures (or) packet losses due to radio channel errors. TCP misinterprets non-congested packet losses (radio channel errors and link failures) as an indication of network congestion and falsely initiate congestion control process which results in the weaker throughput performance.

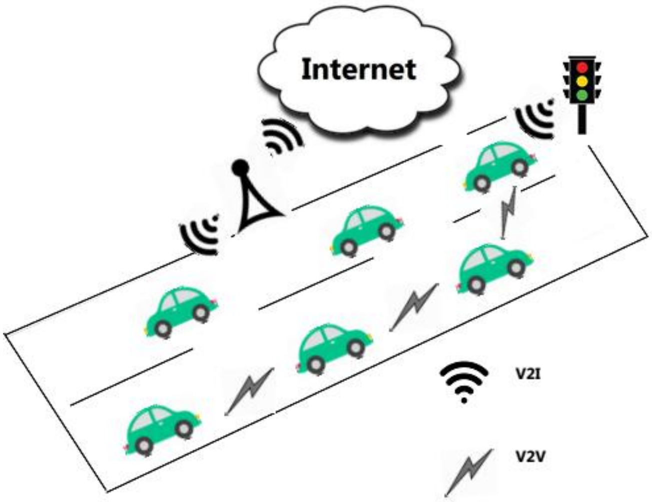


Fig. 1 VANET architecture for V2I and V2V scenarios

TCP [3], a reliable connection oriented layer 4 (L4) transport protocol establishes a virtual connection between the end-to-end process. TCP uses flow control, congestion control, and error control mechanism which guarantees a high-quality data sharing performance in the standard wired network. TCP's congestion control mechanism allows the sender to maintain the sending rate below the network capacity based on the round trip time (RTT) delay or congestion notification from the connecting devices. TCP maintains a state variable for an individual connection, called the congestion window (*cwnd*), which restricts the sending rate based on the feedback from acknowledgment packets (or) congestion notification.TCP remains the largest carrier of internet data traffic due to its inherent capability in error detection and maintaining the packet transmission rate below the receiver advertised capacity and the network capacity.

The congestion control algorithm governs the throughput and fairness performance of TCP flows. The congestion control algorithm is grouped into the network-assisted and independent approaches. In the network-assisted approaches, the sender initiates the transmission rate control based on the congestion warning messages from the intermediate connecting devices. On the other hand, independent approaches are proactive nature, which relies on computing current RTT value with the predefined congestion threshold ($Cong_{Thresh}$) parameter. TCP assumes congestion when the current RTT value is larger than the $Cong_{Thresh}$ and vice versa.

In the Vehicular environment, independent congestion approach performances are compromised by frequent link failure and radio propagation impacts which arises due to the vehicle mobility. The following limitations severely deteriorate the throughput performances of the independent congestion control algorithms.

Spurious rate reduction: During link failures, intermediate vehicular nodes re-route the packets through a new route, which results in the abrupt increase in RTT value. TCP's $Cong_{Thresh}$ parameter misinterprets increase in RTT value as an indication of network congestion and falsely initiate the rate control process. The spurious rate reduction phenomena due to weakly bounded $Cong_{Thresh}$ parameter results in severe throughput deterioration.

Transmission rate halving: On detecting network congestion, the sender reduces the packet rate by half of the *cwnd* irrespective of TCP flow's window growth level. The AIMD algorithm's *cwnd* halving worsens the fairness performance of TCP flows with different window growth levels.

In this paper, a much improved TCP known as vehicular customized TCP (VC-TCP) is proposed to outflank the existing limitations of the independent congestion control approach in the vehicular environment. VC-TCP implements window growth level based transmission rate control in the congestion control process and faster window recovery mechanism in the CR phase, which regulates the transmission rate halving impact and improves the flow fairness in the recovery phase. Furthermore, VC-TCP implements a more durable $Cong_{Thresh}$ parameter based on average RTT, which minimizes the spurious rate reduction limitations due to sudden RTT spike.

The subdivisions of this article are organized in the following order. Section 2 reviews the recently proposed independent congestion control approaches. Section 3 briefs about the proposed VC-TCP congestion control approach. Section 4 compares the vehicular simulation results and its performance analysis under varying traffic and hop length conditions. Section 5 binds up with the proposed VC-TCP's conclusion.

II. Related Works

A vast majority of the protocols [4,5,6] were developed to address the routing-related problems in vehicular networks. However, very few approaches [7,8] were proposed to solve network congestion in vehicular communication networks. This literature review primarily discusses the prominent Vegas-based independent congestion control approach and their enhancements from its inception. TCP Vegas [9] congestion control algorithm aims at improving end-to-end packet latency performance by implementing an RTT based rate reduction approach. Vegas congestion avoidance algorithm adjusts the packet sending rate based on the difference value or $Cong_{Thresh}$ (Δ) by comparing actual RTT with the expected RTT.

$$\Delta = (Expected - Actual) * BaseRTT \tag{1}$$

$$Expected = \frac{cwnd}{BaseRTT} \tag{2}$$

$$Actual = \frac{cwnd}{RTT_i} \tag{3}$$

Vegas defines three thresholds (γ, α, and β), which determines the *cwnd* adjustment rates. In the SS phase, the sender compares Δ with γ and fix the sending rate for each RTT. In the CA phase, *cwnd* is varied based on comparing the Δ value with α and β parameters. The Δ estimate is computed once per RTT period. Furthermore, Vegas window growth follows an additive increase and additive decrease (AIAD) approach which severely cripples fairness performance.

$$\Delta = \begin{cases} cwnd + \dfrac{1}{cwnd}, & if\ \Delta \le \alpha \\ cwnd, & if\ \alpha < \Delta < \beta \\ cwnd - \dfrac{1}{cwnd}, & if\ \Delta \ge \beta \end{cases} \tag{4}$$

Vegas Plus [10] modifies the RTT based congestion threshold parameter and the *cwnd* growth function of the traditional Vegas algorithm. In addition to RTT computation, Vegas Plus algorithm computes a new variable called virtual queue occupancy (VQO), which is predicted from the queuing delay and queue service rate of the bottleneck link. The queuing delay is computed by taking the difference value of current RTT with the minimum RTT value. The queue service rate is calculated from the reception rate of ACK packets.

Enhanced Vegas [11] approach fixes the base RTT value based on the following time durations, forward fixed delay time, forward queuing time, backward fixed delay time and backward queuing time. The forward fixed delay gives the packet latency (propagation delay and packet processing time) from source to destination, and the backward fixed delay is the vice-versa. The forward and backward queuing time is estimated by using the TCP timestamp option in the header field. Enhanced Vegas improves the prediction accuracy and throughput performance than the traditional approach.

Vegas-ad-hoc [12], is developed to address the false Δ value which arises due to frequent route failure associated with high mobility conditions. In the distributed wireless ad-hoc network, the path information is updated frequently due to link failures. The sender fails to update the base RTT value for the new path which results in spurious Δ value and frequent initiation of needless rate reduction. Vegas-ad-hoc uses explicit link failure notification by adding a route change notification (RCN) in the IP and TCP headers. The sender updates the base RTT value on reception of the RCN in the ACK packet, which improves the Δ computation and *cwnd* rate adjustments.

Vegas-W [13] is the modification of traditional algorithm for multi-hop wireless ad-hoc networks. In Vegas-W, *cwnd* is updated after the reception all ACK packets. In case of missing ACK, the sender keeps the *cwnd* as constant and update the sending rate after the reception of all ACK packets. Furthermore, Vegas-W fix the new slow start threshold (SS_{Thresh}) value as 0.875 instead of 0.5.

Pegas [14] and Gvegas[15] algorithms were developed to address spurious rate reduction limitations, which arises due to re-routing. During re-routing, Vegas algorithm fails to update the new base RTT value for the updated routes, which results in needless transmission rate reduction. Pegas algorithm implements a particle swarm optimization approach to update the route change in the network. Similarly, Gvegas executes grey prediction theory to update the base RTT value during the route updates. Apart from controlling the transmission rate at the source node, the Vegas algorithm is applied to adjust queue levels of intermediate devices or routers [16]. In this approach, a queue controller is deployed at the routers to modify the queue levels based on the congestion control parameters (α and β).

Dynamic TCP-Vegas [17] algorithm is designed to address the core issue of fixing an adjustable base RTT value, SS_{Thresh} and dynamically adjusting the sending rate in the CA phase. Dynamic TCP-Vegas uses cuckoo search algorithm to determine the new base RTT value for every timeout (or) route change conditions. The dynamic TCP-Vegas estimates the SS_{Thresh} based on available bandwidth with Δ value. In the CA phase, dynamic TCP-Vegas follows the new growth pattern instead of a fixed increment rate as given in equation (4). Despite the improvements in independent congestion control approaches, these algorithms fail to address the limitations such as flow fairness, spurious rate reduction, and transmission rate halving. The succeeding section discusses the VC-TCP implementation in a vehicular environment which considerably improves flow fairness, throughput stability and overcomes the inadequacies of the existing independent congestion control approaches.

III. PROPOSED VC-TCP APPROACH

The proposed VC-TCP requires modification in $Cong_{Thresh}$ value, rate control process and recovery algorithm. The VC-TCP is of a proactive nature, which initiates rate control process based on the $Cong_{Thresh}$ value using an RTT prediction method. During the data transfer phase, the TCP sender sets a SS_{Thresh} value or maximum window size (*mwnd*) based on the receiver advertised window size (*awnd*). The *awnd* gives the maximum buffer capacity required by the receiving process to handle the packets. The proposed VC-TCP algorithm introduces a new parameter known as window utilization threshold (WUT) value, which is the one half of initial SS_{Thresh} value, i.e. 50% of receiver's maximum capacity. The sender initiates data transfer and doubles the *cwnd* size for each RTT until the SS_{Thresh}. The sender initiates CA phase, when the window growth level exceeds SS_{Thresh} value and the *cwnd* size is incremented one packet per RTT. During each RTT, the sender compares the recent RTT value with $Cong_{Thresh}$ (Δ) and initiate rate reduction, when the current RTT exceeds Δ value. Based on the repeated trials, the proposed VC-TCP implements a $Cong_{Thresh}$ (Δ) value as

$$CongThresh\,(\Delta) = (min_{rtt} + (max_{rtt} - minr_{tt})*\gamma) \tag{5}$$

Where min_{rtt} is the minimum RTT observed and max_{rtt} is the maximum RTT

Condition I: when Δ > recent RTT → No congestion

Condition II: when Δ < recent RTT → Congestion, initiate rate reduction

When the sender experience timeout (or) 3DUPACK, sender retransmit the lost packet and verifies the congested conditions (II) for packet loss. In the case of congested packet loss, sender initiates rate reduction based on the TCP flow's utility level, i.e., WUT value determines the *cwnd* window utility level of the flow. Based on the WUT, the sender computes two distinct rate reduction approach for the flows with higher and lower utilization levels. The growth-based rate reduction approach improves flow fairness performance of different RTT flows.

Condition III: when *cwnd* > WUT → High bandwidth flow, the new SS_{Thresh} value is fixed as

$$SS_{Thresh} = cwnd*(1-\alpha) \tag{6}$$

Where α (α =½) is the window decrement factor for high bandwidth flow.

Condition IV: when cwnd < WUT → Low bandwidth flow, the new SSThresh value is fixed as

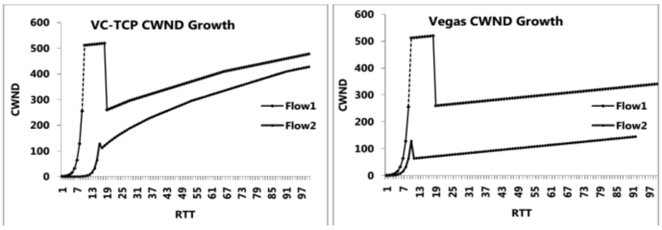
$$SS_{Thresh} = cwnd*(1-\beta) \tag{7}$$

where β ($\beta = \frac{1}{8}$) is the window decrement factor for high bandwidth flow [13].

The proposed VC-TCP rate reduction approach uses different proportion to reduce the transmission rate based on the window utility. The proportion rate reduction mechanism enhances fairness among TCP flows with different RTT. In congestion CR phase, the traditional approach increments its *cwnd* size of one packet per RTT, which results in slower flow convergence to the steady-state point. On the other hand, the proposed VC-TCP implements a newer recovery algorithm with an improved linear trajectory which replaces traditional recovery algorithm with improved flow fairness and throughput stability. The *cwnd* is updated in the CR phase as

$$cwnd_{i+1} = cwnd_i + \frac{2*mwnd}{cwnd} \tag{8}$$

Fig 2. shows the *cwnd* growth of proposed vs. traditional approach. The graph shows the window growth function of two flows for 100 RTTs. The proposed VC-TCP's congestion recovery algorithm narrows down the gap between flow 1 and flow 2, resulting in better flow fairness. On the other hand, flow 1 and flow 2 of traditional approach remain unfair in sharing bandwidth among themselves, i.e., large separation is maintained between flow1 and2. Furthermore, the VC-TCP's rate reduction approach based on the *cwnd* growth improves TCP flows fairness than the traditional approach.



(a) (b)

Fig. 2. window growth function of proposed and existing approach

In the subsequent section, the performance of the proposed VC-TCP algorithm is validated against the Vegas algorithm under vehicular network.

IV. Simulation Results and Analysis

The performance of the proposed VC-TCP is validated against the existing Vegas congestion control approach under multi-hop vehicular wireless environment using network simulator (NS-2) [18]. The simulation experiments were performed for 1000 seconds under urban, and highway scenarios. The mobility model for the analysis were created using VanetMobisim simulator [19]. Fig. 3 shows the node (or) vehicular mobility scenario created using VanetMobisim simulator.

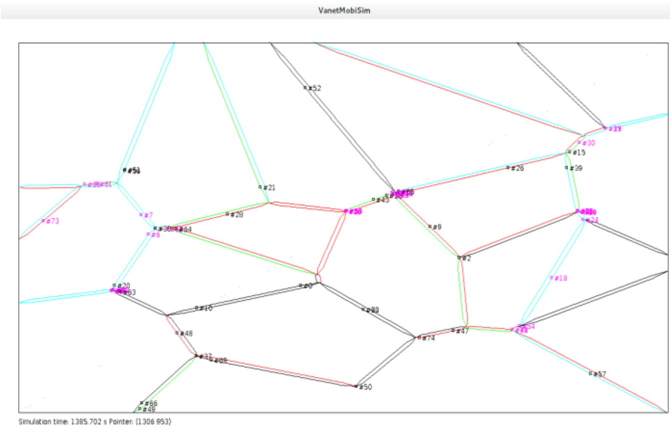


Fig. 3 Mobility scenario generated using VanetMobisim simulator

The performances of the proposed and existing congestion control approaches are validated using as throughput, goodput, end-to-end packet latency and flow fairness. Table 1 gives more details on the simulation parameters.

TABLE I. Simulation Parameters

Simulation Parameters	
Area	2000 x 2000 m
MAC	IEEE 802.11p
Frequency	5.18 GHz
Antenna type	Omni directional
Antenna height	1.5m
Transmitter Antenna gain	1
Receiver Antenna gain	1
Capture Threshold	10 db
Propagation Model	Nakagami
Simulation Time	1000s
Routing Protocol	AODV
Traffic	2, 4, 8
No. of Nodes	100
Window Size	64
TCP Packet Size	1024

The primary analysis is carried out to measure the performance of the congestion control algorithm under different traffic condition. The traffic rate for the analysis is varied between 2,4,6 and 8 traffic flows. Fig. 4 shows the graphical representation of throughput, goodput, end-to-end packet latency and flow fairness performances of the existing and proposed approaches under varying traffic conditions.

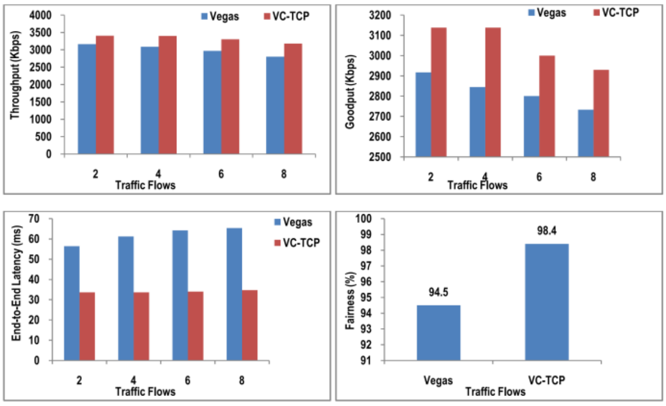


Fig. 4 (a). Traffic flows vs. Throughput (Kbps), (b) Traffic flows vs. Goodput (Kbps), (c) Traffic flows vs. end-to-end latency (ms) and (d) Traffic flows vs. Fairness (%)

Fig. 4 (a) shows the throughput value attained by the proposed and existing approaches under varying traffic conditions. The throughput of VC-TCP achieves a 12% improvement over the existing Vegas approach. The VC-TCP's newer congestion threshold parameter limits the spurious rate reduction which results in improved throughput stability. On the other hand, Vegas congestion threshold parameter fails to endure an abrupt increase in RTT spike value which results in repeated spurious rate reduction. Fig. 4 (b) shows the goodput performance of the proposed and existing approach. The goodput of VC-TCP attains a 10% improvement than the existing Vegas approach. VC-TCP's congestion threshold parameter and the newer window recovery algorithm allows the flow to maintain much improved goodput than its counterpart.

Fig. 4 (c) shows the reduction in end-to-end packet latency performance of the proposed VC-TCP over existing approach. The end-to-end delay of the proposed VC-TCP attains 47% lower than the existing Vegas due to its improved congestion prediction mechanism and faster responsiveness in delivering data packets. Fig. 4 (d) shows the flow fairness performance of proposed and the existing approaches. The proposed VC-TCP congestion control algorithm attains 98.4% flow fairness due to growth based *cwnd* reduction and faster recovery growth algorithm in the CR phase. During the recovery process, the proposed VC-TCP's faster recovery growth algorithm allows the low rate flow to utilize more bandwidth than the high rate flow, which results in achieving flow fairness among high and low rate flows. However, the existing Vegas attain 94.5% flow fairness among two different RTT flows.

A second analysis validates VC-TCP's performances under different vehicle mobility conditions. For this analysis, an urban scenario with the mobility of 15 m/s and highway (freeway) scenario with a mobility of 25 m/s were created. Fig. 5 shows the graphical representation of throughput, goodput, end-to-end packet latency and flow fairness performances of the existing and proposed approaches under different vehicular mobility scenarios.

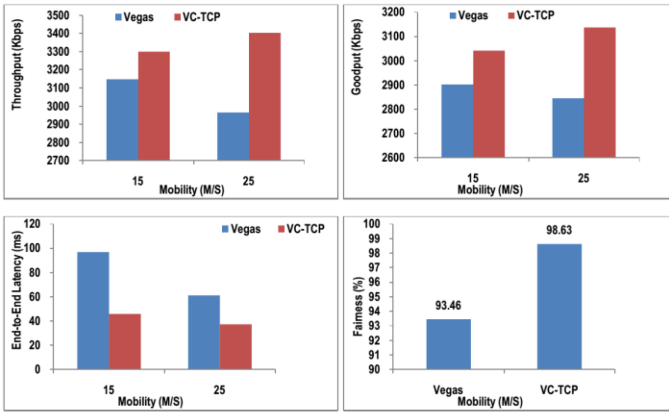


Fig. 5 (a). Mobility (m/s) vs. Throughput (Kbps), (b) Mobility (m/s) vs. Goodput (Kbps), (c) Mobility (m/s) vs. end-to-end latency (ms) and (d) Mobility (m/s) vs. Fairness (%)

Fig. 5 (a) shows the comparative analysis of throughput value attained by the proposed and existing congestion control algorithms under different vehicular mobility scenarios. The proposed VC-TCP approach yields a 12% improvement in throughput value than the existing Vegas approach. During node mobility, there is a likelihood of path breakage and re-routing of packets in the newly established route results in RTT spike. The existing Vegas congestion threshold parameter misinterprets an abrupt increase in RTT spike value as congestion and initiates spurious rate reduction, results in throughput degradation. On the other hand, the proposed VC-TCP's newer congestion threshold parameter can withstand the abrupt increase in RTT spike thereby minimizes spurious rate reduction, which results in improved throughput stability.

Fig. 5 (b) shows the goodput performance of the proposed and existing approaches. The goodput of VC-TCP attains a 10% improvement than the existing Vegas approach. VC-TCP's congestion prediction method and its newer window recovery algorithm allow the TCP flow to attain a much improved goodput than its counterpart. Fig. 5 (c) shows the reduction in end-to-end packet latency performance of the proposed VC-TCP over existing approach. The proposed VC-TCP attains 47% lower end-to-end packet latency than the existing Vegas due to its improved congestion prediction mechanism and faster responsiveness in delivering data packets. Fig. 5 (d) shows the flow fairness performance of proposed and the existing approaches. The proposed VC-TCP congestion control algorithm attains 98.63% flow fairness due to growth based *cwnd* reduction and faster recovery growth algorithm in the CR phase. During the recovery process, the proposed VC-TCP's faster recovery growth algorithm allows the low rate flow to utilize more bandwidth than the high rate flow, which results in achieving flow fairness among high and low rate flows. However, the existing Vegas attain 93.46% flow fairness among two different RTT flows.

V. CONCLUSION

Fairness and throughput performance of independent congestion control approaches under inter-vehicular networks are weakened by the limitations of flat transmission rate reduction in the congestion control process, the slower window recovery rate in the congestion recovery phase, and spurious rate reduction due to weakly shaped congestion threshold parameter. VC-TCP presented in this paper implements a transmission rate reduction based on the sender's bandwidth utilization which significantly improves fairness performances of TCP flows. In the recovery phase, VC-TCP executes a novel window growth function which enhances the transmission rate compared with the current RTT delay based approaches. Furthermore, VC-TCP introduces a much improved RTT delay criterion based congestion threshold parameter which helps the sender to minimize spurious rate reduction impact during non-congested packet drops. Three standard metrics were chosen to validate the performance of the VC-TCP algorithm under inter vehicular environment. The simulation results prove that VC-TCP achieves a significant reduction in end-to-end packet latency (47% against the Vegas), improved throughput and goodput rate (10% and 10% against the Vegas) against the existing approach under inter vehicular environment.

ACKNOWLEDGMENT

A heartfelt thanks to Self Organizing Network Group (SONG) of ECE department, Kongu Engineering College for their encouraging support and valuable guidance in completing this project. We also thank the unknown reviewers for their valuable suggestions in reviewing this article.

REFERENCES

[1] Zeadally, S., Hunt, R., Chen, Y.S., Irwin, A. and Hassan, A., "Vehicular ad hoc networks (VANETS): status, results, and challenges", Telecommunication Systems, vol. 50, no. 4, pp.217-241, Aug. 2012.

[2] Gerla, M. and Kleinrock, L., "Vehicular networks and the future of the mobile internet", Computer Networks, vol. 55, no. 2, pp.457-469, Feb. 2011.

[3] Postel, J., "Transmission Control Protocol", RFC 793, IETF Network Working Group, 1981.

[4] Ramakrishnan, B., Nishanth, R.B., Joe, M.M. and Selvi, M., " Cluster based emergency message broadcasting technique for vehicular ad hoc network", Wireless Networks, vol. 23, no. 1, pp. 233-248, Jan. 2017.

[5] Oliveira, R., Montez, C., Boukerche, A. and Wangham, M.S., "Reliable data dissemination protocol for VANET traffic safety applications", Ad Hoc Networks, 63, pp.30-44, Aug. 2017.

[6] Karimi, R. and Shokrollahi, S., "PGRP: Predictive geographic routing protocol for VANETs", Computer Networks, vol. 141, pp.67-81, Aug. 2018.

[7] Jude, M.J.A., Diniesh, V.C., Shivaranjani, M. and Shanju, R., "A Feedback Aware Reliable Transport Protocol with Improved Window Increment Mechanism for Inter Vehicular Wireless Network", Wireless Personal Communications, vol. 98, no. 1, pp.1119-1134, Jan. 2018.

[8] Qureshi KN, Abdullah AH, Kaiwartya O, Iqbal S, Butt RA, Bashir F, "A Dynamic Congestion Control Scheme for safety applications in vehicular ad hoc networks", Computers & Electrical Engineering, Dec. 2017.

[9] Lawrence S. Brakmo, and Larry L. Peterson, 'TCP Vegas: End to End Congestion Avoidance on a Global Internet,' IEEE Journal on Selected Areas in Communications, vol. 13, no. 8. pp.1465-1480, Oct 1995.

[10] Chen, J.R. and Chen, Y.C., "Vegas plus: Improving the service fairness", IEEE Communications Letters, vol. 4, no. 5, pp.176-178, May 2000.

[11] Chan, Y.C., Chan, C.T. and Chen, Y.C., "An enhanced congestion avoidance mechanism for TCP Vegas", IEEE Communications Letters, vol. 7, no. 7, pp.343-345, Jul. 2003.

[12] Ding, L., Wang, X., Xu, Y. and Zhang, W., "Improve throughput of TCP-Vegas in multihop ad hoc networks", Computer Communications, vol. 31, no. 10, pp.2581-2588, Jun. 2008.

[13] Kim, D., Bae, H. and Toh, C.K., "Improving TCP-Vegas performance over MANET routing protocols," IEEE transactions on Vehicular Technology, vol. 56, no. 1, pp.372-377 , Jan. 2007.

[14] Jamali, S., Alipasandi, N. and Alipasandi, B., "TCP pegas: A PSO-based improvement over TCP Vegas", Applied Soft Computing, vol. 32, pp.164-174, Jul. 2017.

[15] Jiang, H., Luo, Y., Zhang, Q., Yin, M. and Wu, C., "TCP-Gvegas with prediction and adaptation in multi-hop ad hoc networks", Wireless Networks, vol. 23, no. 5, pp.1535-1548, Jul. 2017.

[16] Abolfazli, E. and Shah-Mansouri, V., "Dynamic adjustment of queue levels in TCP Vegas-based networks", Electronics Letters, vol. 52, no. 5, pp.361-363, Feb. 2016.

[17] Sunitha, D., Nagaraju, A. and Narsimha, G., "Dynamic TCP-Vegas based on cuckoo search for efficient congestion control for MANET", International Journal of Signal and Imaging Systems Engineering, vol. 10, no. 1-2, pp.47-53, 2017.

[18] NS Simulator, http://www.isi.edu/nsnam /ns

[19] VanetMobisim,http://vanet.eurecom.fr.