Advances in Nonlinear Variational Inequalities
ISSN: 1092-910X
Vol 28 No. 2 (2025)

Adaptive QoS-aware Congestion Window Adaptation through Contention Detection in Wireless Networks

Priya Kumari[1], Dr.Nitin Jain[2]
[1]Research Scholar, Dept of ECE, Babu Banarasi Das University, Lucknow, UP. priya.sinha942@gmail.com
[2]Prof and Head of Dept of ECE, Babu Banarasi Das University, Lucknow, UP. hod.ec@bbdu.ac.in
Corresponding Author: Priya Kumari

Article History:

Received: *30-07-2024*

Revised: *13-09-2024*

Accepted: *21-09-2024*

Abstract:

In wireless networks, maintaining Quality of Service (QoS) while managing congestion effectively is a critical challenge due to changing network conditions. Quality of service (QoS) technique operates on a network to run high-priority applications and traffic reliably despite limited network capacity. This paper proposes an adaptive QoS-aware congestion window adaptation mechanism that leverages contention detection to optimize performance in wireless networks. A cooperative relay node has also been introduced to relay safety messages that failed to reach the destination. According to the proposed method, congestion window size is dynamically adjusted to maintain QoS standards even for high-priority traffic under varying network conditions. Simulation results obtained for our model show that using a dynamic window enhances the performance of the system by allowing higher throughput, stable performance, a decrease in delay, and a higher packet success rate.

Keywords: Wireless Networks, Quality of Service.

1 Introduction

In the early days of wired networks, Transmission Control Protocol (TCP) [1] was developed for reliable acknowledgement-based transport layer protocols. As TCP's early days were plagued by congestion problems, congestion algorithms [2], [3] were incorporated for further development. As a result of its reliability, it is widely used in Internet applications, including email, remote access, and file transfers. A TCP connection can carry up to 90 percent of all internet traffic [4], [5], making it one of the most important protocols. Besides wireless Internet access, wireless ad-hoc networks (WANETs) have also emerged as a result of wireless technology. There is a benefit to these networks when there is no infrastructure available or when the infrastructure is expensive. The WANET is composed of wireless nodes, which communicate wirelessly without using access points. Nodes act as hosts and routers. Easily deployable anywhere, anytime. This alluring paradigm may lead to the emergence of new emerging networks, such as the Internet of Things, wireless sensor networks, vehicular ad hoc networks, and tactical wireless networks [5], [6]. IEEE 802.11 is the de facto medium access standard for WANETs [7].

Medium Access Control (MAC) protocol plays a significant role in energy conservation in wireless networks. For packet collision prevention, the MAC protocol controls which node can access the shared wireless medium at any given moment. A node that is not entitled to occupy a medium is forced to sleep. In the MAC protocol, channels are allocated and occupied according to duty-cycle algorithms [8]. MAC protocols have addressed a number of WSN-related issues, including ineffective channel utilization and unnecessarily high energy consumption [9]. Contending-based

MAC strategies are most popular because they are simple and efficient. In addition, sensor-based applications can make use of several wireless routing protocols. A routing protocol identifies and establishes a multi-hop data transmission path between two nodes [10]. When compared with single-layer protocols, cross-layer protocols perform better overall [11], [12], [13], [14]. Layered protocols do not allow cross-layer interaction, but cross-layer design does [15]. An energy-efficient approach is developed in [16] by integrating the network, data link, and physical layers. There is often a trade-off between reliability and latency when routing is done at the single layer or vice versa.

Packet-switched networks face congestion issues. Whenever a network's load exceeds its capacity, overload occurs. As a result, there are techniques for handling congestion. These techniques can maintain a load below capacity. In addition to the systems that are involved in waiting, the routers and switches may also be congested. The performance of a network is measured by two factors: delay and throughput. A small load will result in minimal network delay, a large load will cause network delays to increase suddenly, and an infinite load will result in infinite delays. As a result of delays, load is reduced, resulting in congestion. As a measure of throughput, bits pass through a point in a second. When the network is under load, throughput is high. However, when the load reaches a certain limit, the traffic drops sharply because the routers discard the packets.

2 Literature Review

In [17], There are two protocols that WCCP uses to control congestion: SCAP (source congestion avoidance protocol), which controls congestion at source nodes, and RCCP (receiver congestion control protocol), which controls congestion at intermediate nodes. The transmission rate is adjusted in SCAP based on inter-departure time (PIDT). In RCCP, congestion is detected by measuring the queue lengths of intermediate nodes. As a result of RCCP congestion, SCAP is notified. This problem is later fixed by SCAP, which adjusts the PIDT and sends only I-frames to intermediate nodes, ignoring the P-frames and B-frames. A new PIDT is applied by SCAP, P and B packets (less important packets) are ignored, and only I frames are sent to intermediate nodes when such conditions exist. During congestion, neighbours are notified, and local rate adjustments are made to prevent congestion from spreading. As a result, the neighbors' sending rates decline. Depending on the local network conditions, back-pressure propagation upstream is determined. Congestion propagates up to the sources if it persists. Sources request constant feedback from sinks via the regulated bit within event packets in order to maintain their rate. Sources reduce their rate if they do not receive ACKs.

Additionally, the sink stops sending ACKs when it receives low rates of event packets from the source. There are no provisions to ensure fairness, and the scheme does not guide how to change the rate after congestion has occurred. In the absence of well-designed traffic control, bandwidth may be wasted.

As a distributed and scalable algorithm, CCF has been initially proposed in [19] as a means to eliminate network congestion in sensor networks and to ensure packet delivery to sinks fairly. A CCF-based data-link transport layer works with any MAC protocol. CCF algorithms measure the average packet rate from each node, divide it among its children, adjust the rate if queues are about to overflow, and propagate it downstream. As a result of using packet service time, CCF can calculate how much service is available. According to the number of child nodes and service rates available at each node, congestion is controlled hop-by-hop. CCF ensures simple fairness, as can be demonstrated. Here are two major problems with CCF, as shown in [18]. In cases where some sensor nodes do not have enough traffic or packets are corrupted, CCF's rate adjustment may result in low utilization. A disadvantage of this method is that it has a low throughput when some nodes lack packets to send because of its work-conservation scheduling algorithms.

IEEE 802.11 has been subjected to a number of studies to determine the best value for CW in order to improve the performance of the network. Several authors have proposed a scheme to counter BEB

Multiplicative Increase and Linear Decrease (MILD), which multiplies the CW by 1.5 in the event of a collision and decreases it by 1 when a transmission succeeds [19]. Overhearing successful transmissions is made possible by the CW copy mechanism. After a successful transmission, the transmission is prevented from escalating to a minimum CW automatically, thus improving the transmission quality. Authors of [20] and its authors, exponential increase and exponential decrease (EIED), have been used to improve the performance of a wireless region when there are multiple competing nodes to access the medium. In the wake of a collision, EIED doubled its contention window, and in the wake of a successful transmission, it halved it. LMIDDs are linear, and multiplication increases and linear decreases are proposed in [21]. Collisions increase CW multiplicatively for the sender. CW is set linearly by any node that overhears a collision, and it is decreased linearly by any node that succeeds.

3 Methodology

ECWA-CD is a new method for detecting congestion windows that is efficient and effective. This section introduces a new parameter called the variance of contention RTT per hop (VCRTT-H) and then explains why it is useful for estimating MAC contention. Our approach to solving the overshooting problem is based on this parameter.

As a first step, packet forwarding is used to determine the contention RTT. As packets arrive at the front of the queue, contention delay is determined by how long it takes for those packets to reach the physical layer. Contention delays in the currently connected node are tracked with time stamps added to fragments (frames). Each hop produces a fragment of contention delay, which is copied into the acknowledgement message by the TCP destination. Developing a simulation of contention delays is simplified by turning off the delayed ACK function and turning off the fragmentation function. Therefore, MAC agents were able to copy contention delays directly from fragments to packets. Real-world implementations allowed the MAC agent to select and copy contention samples from packets at the end of fragmentation. As a result of the ACK, the sample value of contention delay is recorded, and the TCP source calculates the contention response time based on that value. Further, the variance of contention RTT (VCRTT) can be determined at a source node, which represents the deviation of contention RTT from the most recent n segments. Calculating VCRTT-H requires dividing the VCRTT by the number of hops in the route.

Our decision to use the VCRTT to measure link contention comes from three reasons. In the first place, it can indicate how much contention there is between links. When channel contention occurs, the MAC contention window is wider, and the backoff time is selected in a wider range; this means that a higher contention probability will result in a different contention delay for each channel. This jittery is measured by variance, which may indicate contention directly. Varying is a way of describing the contention observed during recent observations. It is also possible to determine the status of a bottleneck by examining the variance. A path with a bottleneck will have a larger VCRTT than the other path. As a result of severe contention, a bottleneck link will likely fail. Calculating the variance and dividing it by the hop count after we receive the contention RTT allows us to determine the variance. Once each hop has been calculated, the deviation can be calculated. In addition, instead of taking into account the forward path as in the conventional method [15], we take into account the round-trip path, as the ACKs in the backward route can also be used to detect congested links.

Last but not least, we have to select a suitable value for n, the number of samples required to estimate the VCRTT. Node memory size and calculation cost increase if n is large. Long-term average effects, however, slow its response to changes. Conversely, a small change in n can lead to a large difference in VCRTT-H on different RTTs if n is very small. Following are the equations we use in the implementation to update the VCR once the source receives a new ACK:

$$E_k = \frac{E_{k-1} \times * \, k - 1) + CR_{new}}{k} \; for\; k < n \qquad (1)$$

$$V_k = \frac{\left(V_{k-1} + E_{k-1}^2\right) \times (k-1) + CR_{new}^2}{k} - E_k^2 \; for\; k < n \qquad (2)$$

$$E_k = E_{k-1} + \frac{CR_{new} - CR_{old}}{n} for\; k >= n \qquad (3)$$

$$V_k = \frac{\left(V_{k-1} + E_{k-1}^2\right) \times n + CR_{new}^2 - CR_{old}^2}{n} - E_k^2 \; for\; k >= n \qquad (4)$$

The parameter k represents the sampling sequence of newly received ACKs, n represents the maximum number of samples to be collected, E_k and V_k represent respectively the expectations and variances of CR (contention RTT) based on the latest k or ACKs (if $k > n$). A new CR value is obtained, and a previous CR value is retained. Since $k < n$, is different from n, it follows that V_k is determined by (1) and (2). These equations are calculated after a TCP connection is initiated or a timeout occurs. There is no need for CR_{old} in this case. From (3) and (4), $k >= n, V_k$. A new ACK pushes old data out of the VCR so the VCR can be updated using the new CR value. Divide V_k by the hops when we have just obtained V_k to obtain VCRTT-H.

TCP Congestion Window Adaptation Mechanism

In the network, a severe degree of contention is considered when the VCRTT-H exceeds its threshold parameter. The TCP source updates the VCRTT-H after receiving a new ACK before determining if it is greater than the previous value. As a result, cwnd is reduced by one MSS rather, and the congestion window is increased when it is larger. This approach could effectively prevent congestion window overshoots. Prior to halving the congestion window threshold $ssthresh$, we should check whether the RTO expires, indicating bad congestion or contention on the network. A larger VCRTT-H indicates a very poor network status, and the connection is reset two times to MSS, and slow-start is enabled. Other than that, there are no problems with the network's contention status, and the total number of connections is just halved. An RTO should reset the sampling of VCRTT-H so a low value obtained before RTO expiration does not cause severe slow starts afterwards.

As opposed to traditional TCP contention control, which relies on outmoded contention information to calculate VCRTT-H, the proposed algorithm eliminates outmoded values before they get incorporated into calculations. Its simplicity and stability make it a better congestion window adaptation mechanism than other mechanisms. A new ACK indicates how the cwnd can be adjusted, and even if the estimation of contention differs from reality, only slight adjustments are made to the cwnd. TCP contention control, on the other hand, resets the cwnd when throughput and contention are worse than expected. As a second step, we determine whether packet loss can be minimized depending on the level of contention. Contention status should not be severe because by halving the congestion window, the large variation of cwnd will be reduced, and throughput will increase. There are several advantages to the proposed algorithm, including its robustness and accuracy.

4 Result Analysis and Discussion

As part of the evaluation, several parameters are evaluated, including packet success rate (PSR), throughput based on link utilization, and variation between congestion windows. Upon rigorous examination of the simulation results, we find that the proposed congestion control approach outperforms TCP-CUBIC, mAIMD, and mSACK with respect to buffer utilization and throughput.
A graph showing the number of packets that were successfully sent at different RTTs is shown in Figure 1. The success rate of a project is advanced when there is minimum packet loss. Despite the

lower success rates achieved by the mAIMD approach, all models using the early recovery model achieve close to 99.99% success rates. Congestion avoidance algorithms reduce congestion windows by half at each packet drop because of early recovery models. This results in an underutilization of bandwidth after reducing the window size. TCP-CUBIC and mAIMD have higher success rates (%) than the proposed model.

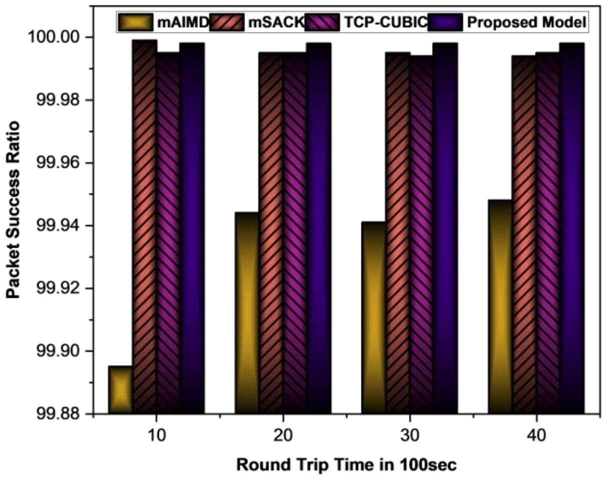

Figure 1: Packet success rate (%) versus round trip time.

As part of our experiments, we also monitor the duration of the congestion window minutely. Figure 2 illustrates how the proposed model maintains consistency with respect to the congestion window length. In previous posts, we discussed how mAIMD and mSACK reduce congested windows by half if a packet is lost. mAIMD and mSACK models do not reduce congestion windows except during the early recovery phase; instead, after reaching the threshold, they increase it linearly or exponentially. During early recovery, existing models show linear increases in congestion windows.

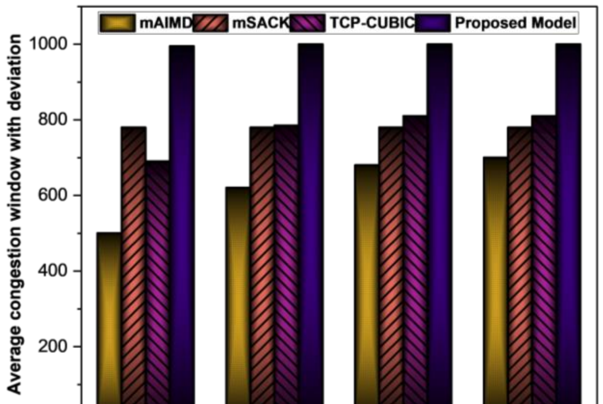

Figure 2: Average congestion window versus round trip time.

We simulated the proposed model to evaluate its performance. In addition to VCAR-MAC [22], RECV-MAC [23], Traditional MAC, and a Clustering model [24], we compared our results with other comparable models. Comparing the proposed model to others, Figure 3 displays the highest throughput. As a result of the proposed model for multi-hop communication, the CW value is determined accurately and the best relays are selected for transmission.
As a result, packet collisions are reduced, resulting in higher throughput. Due to the fixed CW used in the RECV-MAC model, low vehicle density results in a lower throughput. In contrast, packet drop ratios increase as vehicle density increases, resulting in decreased throughput. Due to CW

adjustment, the throughput performance is comparatively smaller when the vehicle density increases in this proposed model.

Compared with other models, the proposed model appears to have a significantly shorter average packet delay in Figure 3. The best relay is selected by a dynamic CW adjustment, which eliminates packet collisions and MAC layer delays. There have been comparisons where the same CW is assumed for all vehicle densities, resulting in higher packet collisions with higher vehicle densities. Due to this, low-density networks are not sufficiently covered by such models. Nevertheless, the proposed model dynamically adjusts the CW before transmission.

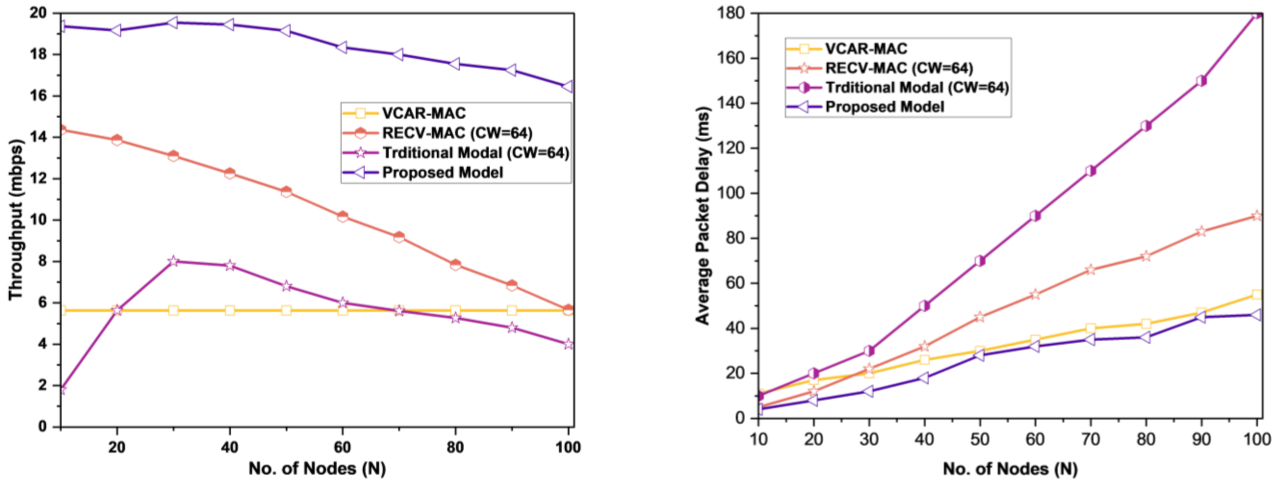

Figure 3: The average throughput and delay versus number of nodes.

5 Conclusion

The paper presents a wireless network contention detection mechanism that implements real-time QoS awareness. With the proposed method, congestion windows are dynamically adjusted based on detected levels of contention, ensuring that high-priority traffic maintains the required Quality of Service. Simulation results showed that our approach improved throughput, reduced latency, and provided a balanced trade-off between efficiency and fairness. The results showed that delay and packet loss were within the boundary of the quality requirement, and the same effect was observed with more than one input source despite constrained bandwidth. The adaptive mechanism we developed offers a more robust and responsive solution to the challenges of managing congestion in wireless environments than traditional congestion control methods. Adaptive congestion control strategies for modern wireless networks are discussed in this work as the foundation for further research.

Reference

[1] J. Postel, "Transmission Control Protocol," RFC Editor, RFC0793, Sep. 1981. doi: 10.17487/rfc0793.

[2] V. Jacobson, "Congestion avoidance and control," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 18, no. 4, pp. 314–329, Aug. 1988, doi: 10.1145/52325.52356.

[3] M. Allman, V. Paxson, and E. Blanton, "TCP Congestion Control," RFC Editor, RFC5681, Sep. 2009. doi: 10.17487/rfc5681.

[4] M. Al Shinwan, L. Abualigah, N. D. Le, C. Kim, and A. M. Khasawneh, "An intelligent long-lived TCP based on real-time traffic regulation," *Multimed. Tools Appl.*, vol. 80, no. 11, pp. 16763–16780, May 2021, doi: 10.1007/s11042-020-08856-z.

[5] M. J. A. Jude, V. C. Diniesh, M. Shivaranjani, S. Madhumitha, V. K. Balaji, and M. Myvizhi, "Improving Fairness and Convergence Efficiency of TCP Traffic in Multi-hop Wireless Networks," *Wirel. Pers. Commun.*, vol. 121, no. 1, pp. 459–485, Nov. 2021, doi: 10.1007/s11277-021-08645-3.

[6] R. Rukaiya, M. U. Farooq, S. A. Khan, F. Hussain, and A. Akhunzada, "CFFD-MAC: A hybrid MAC for collision free full-duplex communication in wireless ad-hoc networks," *IEEE Access*, vol. 9, pp. 35584–35598, 2021.

[7] L. D. Hieu *et al.*, "Improving Fairness in IEEE 802.11 EDCA Ad Hoc Networks Based on Fuzzy Logic," *J. Adv. Comput. Intell. Intell. Inform.*, vol. 24, no. 5, pp. 615–620, Sep. 2020, doi: 10.20965/jaciii.2020.p0615.

[8] O. Gurewitz, M. Shifrin, and E. Dvir, "Data Gathering Techniques in WSN: A Cross-Layer View," *Sensors*, vol. 22, no. 7, p. 2650, Mar. 2022, doi: 10.3390/s22072650.

[9] A. Kumar, M. Zhao, K.-J. Wong, Y. L. Guan, and P. H. J. Chong, "A Comprehensive Study of IoT and WSN MAC Protocols: Research Issues, Challenges and Opportunities," *IEEE Access*, vol. 6, pp. 76228–76262, 2018, doi: 10.1109/ACCESS.2018.2883391.

[10] D. Dhabliya *et al.*, "Energy-Efficient Network Protocols and Resilient Data Transmission Schemes for Wireless Sensor Networks—An Experimental Survey," *Energies*, vol. 15, no. 23, p. 8883, Nov. 2022, doi: 10.3390/en15238883.

[11] L. Van Hoesel, T. Nieberg, Jian Wu, and P. J. M. Havinga, "Prolonging the lifetime of wireless sensor networks by cross-layer interaction," *IEEE Wirel. Commun.*, vol. 11, no. 6, pp. 78–86, Dec. 2004, doi: 10.1109/MWC.2004.1368900.

[12] M. C. Vuran and I. F. Akyildiz, "XLP: A Cross-Layer Protocol for Efficient Communication in Wireless Sensor Networks," *IEEE Trans. Mob. Comput.*, vol. 9, no. 11, pp. 1578–1591, Nov. 2010, doi: 10.1109/TMC.2010.125.

[13] L. D. P. Mendes and J. J.P.C. Rodrigues, "A survey on cross-layer solutions for wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 34, no. 2, pp. 523–534, Mar. 2011, doi: 10.1016/j.jnca.2010.11.009.

[14] S. Jagadeesan and V. Parthasarathy, "Cross-Layer Design in Wireless Sensor Networks," in *Advances in Computer Science, Engineering & Applications*, vol. 166, D. C. Wyld, J. Zizka, and D. Nagamalai, Eds., in Advances in Intelligent and Soft Computing, vol. 166. , Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 283–295. doi: 10.1007/978-3-642-30157-5_29.

[15] S. P and A. Mathew, "Cross layer design with weighted sum approach for extending device sustainability in smart cities," *Sustain. Cities Soc.*, vol. 77, p. 103478, Feb. 2022, doi: 10.1016/j.scs.2021.103478.

[16] I. Jemili, D. Ghrab, A. Belghith, and M. Mosbah, "Cross-layer adaptive multipath routing for multimedia Wireless Sensor Networks under duty cycle mode," *Ad Hoc Netw.*, vol. 109, p. 102292, Dec. 2020, doi: 10.1016/j.adhoc.2020.102292.

[17] S. M. Aghdam, M. Khansari, H. R. Rabiee, and M. Salehi, "WCCP: A congestion control protocol for wireless multimedia communication in sensor networks," *Ad Hoc Netw.*, vol. 13, pp. 516–534, 2014.

[18] C. Wang, B. Li, K. Sohraby, M. Daneshmand, and Y. Hu, "Upstream congestion control in wireless sensor networks through cross-layer optimization," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 4, pp. 786–795, 2007.

[19] V. Bharghavan, A. Demers, S. Shenker, and L. Zhang, "MACAW: A media access protocol for wireless LAN's," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 24, no. 4, pp. 212–225, 1994.

[20] N.-O. Song, B.-J. Kwak, J. Song, and M. E. Miller, "Enhancement of IEEE 802.11 distributed coordination function with exponential increase exponential decrease backoff algorithm," in *The 57th IEEE Semiannual Vehicular Technology Conference, 2003. VTC 2003-Spring.*, IEEE, 2003, pp. 2775–2778.

[21] A. Paul and S. Rho, "Probabilistic model for M2M in IoT networking and communication," *Telecommun. Syst.*, vol. 62, pp. 59–66, 2016.

[22] S. Cao and V. C. Lee, "A novel coordinated medium access control scheme for vehicular ad hoc networks in multichannel environment," *IEEE Access*, vol. 7, pp. 84333–84348, 2019.

[23] A. F. M. Shahen Shah, H. Ilhan, and U. Tureli, "RECV-MAC: a novel reliable and efficient cooperative MAC protocol for VANETs," *IET Commun.*, vol. 13, no. 16, pp. 2541–2549, Oct. 2019, doi: 10.1049/iet-com.2018.6171.

[24] H. Wang, R. P. Liu, W. Ni, W. Chen, and I. B. Collings, "VANET modeling and clustering design under practical traffic, channel and mobility conditions," *IEEE Trans. Commun.*, vol. 63, no. 3, pp. 870–881, 2015.