ICT Express 11 (2025) 631–635

Contents lists available at ScienceDirect

ICT Express

journal homepage: www.elsevier.com/locate/icte

MSS-TCP: A congestion control algorithm for boosting TCP performance in mmwave cellular networks

Omar Imhemed Alramli [a,b,*], Zurina Mohd Hanapi [a,*], Mohamed Othman [a,c], Normalia Samian [a], Idawaty Ahmad [a]

[a] Department of Communication Technology and Network, Universiti Putra Malaysia, Serdang, 43400, Selangor State, Malaysia
[b] Telecommunications and Networking Department, Misurata University, Misurata, Libya
[c] Laboratory of Computational Science and Mathematical Physics, Institute for Mathematical Research (INSPEM), Universiti Putra Malaysia, 43400 UPM Serdang, Selangor D.E., Malaysia

ARTICLE INFO

Keywords:
Congestion control algorithm (CCA)
Transmission control protocol (TCP)
Maximum segment size (MSS)
MSS-TCP
MmWave cellular networks

ABSTRACT

The increasing demand for high-speed, low-latency applications, especially with 5G mmWave technology, has led to challenges in TCP performance due to signal blockages, small buffers, and high Packet Error Rates (PERs). Existing congestion control algorithms (CCAs) struggle to fully utilize available bandwidth under these conditions. This paper proposes MSS-TCP, a novel congestion control algorithm designed for mmWave networks. MSS-TCP dynamically adjusts the congestion window (cwnd) based on the maximum segment size (MSS) and round-trip time (RTT), improving bandwidth utilization and congestion adaptability. The simulation results using the ns-3 network simulator show that MSS-TCP outperforms state-of-the-art CCAs, including NewReno, HighSpeed, CUBIC, and Bottleneck Bandwidth and Round-trip propagation time (BBR), and Fuzzy Logic-based (FB-TCP), particularly when the buffer matches the bandwidth-delay product (BDP), achieving a 24.26% to 45.43% improvement in throughput compared to BBR while maintaining low latency. These findings demonstrate that MSS-TCP enhances TCP performance in 5G mmWave networks, making it a promising solution for next-generation wireless communication.

1. Introduction

The demand for high internet speed and low latency has recently surged, propelled by the growing dependence on mobile applications within cellular networks [1]. Numerous network applications, including virtual and augmented reality, necessitate high data rates and minimal latency. Moreover, in the 5G era, cellular networks use mmWave technology to fulfill these speed rate and latency application needs. However, the use of mmWave may adversely impact the performance of TCP in cellular networks due to its limited penetration through obstacles like trees, buildings, and human bodies [1–6]. Consequently, in NLoS conditions, this limitation leads to severe performance degradation of TCP in such networks. The Congestion Control Algorithm (CCA) plays a pivotal role in TCP, significantly impacting the overall performance of these networks. Numerous TCP variants have been developed to enhance TCP performance across diverse conditions [7,8]. TCP often struggles to achieve higher performance and fully utilize the available bandwidth, especially when buffers are small and PERs are high [2,9]. TCP performance degradation stems from the varying characteristics of cellular networks, including signal blockage, mmWave signal attenuation, and error loss leading to throughput reduction and increased latency [10]. Recent studies have highlighted various shortcomings in existing TCP variants when it comes to achieving higher data rates and lower latency in 5G mmWave networks, including cellular networks [2,11–13]. TCP protocols have been proposed such as D-TCP [14], 5G-TCP [4], Yinker-BBR [15], mmS-TCP [5], TCP-FLASH [16], RBBR [17], NexGen D-TCP [14], and Fuzzy Logic-based (FB-TCP) [11].

This paper introduces MSS-TCP, a novel CCA designed to address small buffers and high PER issues in cellular networks and to enhance TCP performance in 5G mmWave cellular networks. The key contributions of this paper are: propose an increment factor mechanism for adjusting the congestion window (cwnd) based on the Maximum Segment Size (MSS) and Round Trip Time (RTT). Furthermore, comprehensive simulation experiments are conducted to compare MSS-TCP with state-of-the-art CCAs, including NewReno [18], HighSpeed [19],

* Corresponding authors at: Department of Communication Technology and Network, Universiti Putra Malaysia, Serdang, 43400, Selangor State, Malaysia.
E-mail addresses: o_alrmli@yahoo.com (O.I. Alramli), zurinamh@upm.edu.my (Z.M. Hanapi), mothman@upm.edu.my (M. Othman), normalia@upm.edu.my (N. Samian), idawaty@upm.edu.my (I. Ahmad).

https://doi.org/10.1016/j.icte.2025.05.005
Received 18 November 2024; Received in revised form 14 April 2025; Accepted 12 May 2025
Available online 26 May 2025

CUBIC [20], Bottleneck Bandwidth and Round-trip propagation time (BBR)v1 [21], and FB-TCP [11]. CUBIC is notably the default CCA for both Linux and Windows systems, while BBR is the default for Google servers [15,22], and FB-TCP is a recently proposed protocol to enhance TCP performance over 5G mmWave cellular networks.

The subsequent sections of this paper are as follows: Section 2 introduces the proposed algorithm "MSS-TCP". Section 3 details the methodology applied for performance evaluation, covering aspects such as experimental setup, network topology, simulation parameters, results, and discussion. Lastly, Section 5 concludes the paper and suggests potential directions for future research.

2. MSS-TCP: The proposed algorithm

This section demonstrates the MSS-TCP mechanism and its behavior. The following subsections provide a detailed explanation of the proposed algorithm.

2.1. MSS-TCP mechanism

In the CA phase of TCP's CCA, the cwnd is incremented by an increment value and depends on a factor's value. This factor varies from one TCP to another TCP, as in Eq. (1). For instance, in NewReno, the factor is a predefined constant equal to 1. In CUBIC, the factor is a cubic function. For more instances, in NewReno, the congestion window is increased by $1/cwnd$. On the other hand, in CUBIC, the increment value is determined as $C * (\Delta - \sqrt[3]{(\beta * cwnd)/C}$, where C is a preset constant, β is the multiplicative decrease factor, and Δ represents the elapsed time since the last loss. MSS-TCP increases its cwnd during the CA phase using a novel increment factor based on the MSS and RTT. Unlike existing CCAs, which use predefined constants as in NewReno or cubic functions as in CUBIC, MSS-TCP's factor dynamically adapts to network conditions, forming a convex growth curve. The primary contribution of MSS-TCP is its unique cwnd growth function, as illustrated in the following Eq. (6).

$$cwnd = cwnd + \frac{Factor}{cwnd} \tag{1}$$

As known, MSS is equal to one segment size as Eqs. (2) and (3) show, respectively.

$$One\ segment\ size\ = \frac{segments\ size}{number\ of\ segments} \tag{2}$$

$$MSS = \frac{segments\ size}{number\ of\ segments} \tag{3}$$

RTT variations can be utilized to measure the network's status [7, 22–27]. Therefore, the Time Ratio is set as a function in three different RTTs, as Eq. (4) shows. This Time Ratio has different values according to the network conditions.

$$Time\ Ratio = \frac{(RTT_{max} + RTT_{base})}{RTT_{cur}} \tag{4}$$

where RTT_{cur} is the current RTT, RTT_{base} is the minimum RTT, and RTT_{max} is the maximum RTT.

The magnitude of the maximum segment size is chosen alongside RTT as a parameter for incrementing cwnd during the CA phase. Many extensive simulations have been conducted, and we found the best result for MSS-TCP performance when the increment factor parameter relies on the MSS magnitude value.

MSS-TCP calculates a dynamic parameter, the novel factor, based on network states, as shown in (5). This calculation involves multiplying Eq. (3) with Eq. (4), relying on MSS and Time Ratio values. The resulting value dictates the increment of cwnd accordingly.

$$Factor = \sqrt{MSS * Time\ Ratio} \tag{5}$$

The novel factor should dynamically create an adaptive convex-up curve as Fig. 1 shows, decreasing the epoch time and maximizing the area under the cwnd curve to improve TCP performance in mmWave networks. The choice of a square root function to represent the novel factor is based on fulfilling the adaptive increase requirement and its simplicity and suitability for integration into the core space of the Linux kernel.

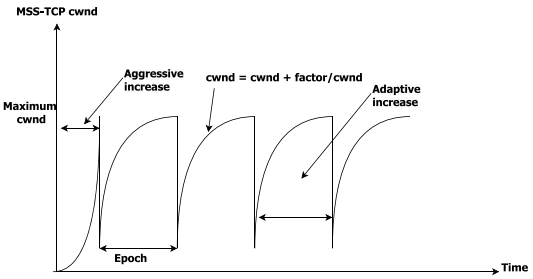


Fig. 1. MSS-CCA cwnd behavior.

Finally, Eq. (1) becomes a complete formula as illustrated in Eq. (6), and Algorithm 1 describes the pseudo-code of the proposed algorithm.

$$cwnd = cwnd + \frac{\sqrt{MSS * Time\ Ratio}}{cwnd} \tag{6}$$

Algorithm 1: MSS-Based CCA

```
Initialize: cwnd, ssthresh, rttbase, rttcur, rttmax;
if there is data to send then
    if RTO not expired then
        if no 3DACK then
            if cwnd < ssthresh then
                cwnd = cwnd + 1;
            else
                rttcur = Acktime − Sendtime;
                if rttbase > rttcur then
                    rttbase = rttcur;
                end
                if rttmax < rttcur then
                    rttmax = rttcur;
                end
                TimeRatio = (rttmax+rttbase) / rttcur ;
                MSS = segmentsize / numberofsegments ;
                Factor = √(MSS × TimeRatio);
                cwnd = cwnd + Factor / cwnd ;
            end
        else
            Multiplicative decrease;
        end
    else
        Slow start stage;
    end
end
```

2.2. MSS-TCP behavior

Like standard TCP, MSS-TCP initiates the slow start stage [28–30], demonstrating an exponential increase by doubling cwnd in each RTT until reaching the slow start threshold (ssth) or the sender receives three Duplicate Acknowledgments (3DACKs). Therefore, the protocol triggers the CA phase, and the cwnd increases gently to prevent the network from congestion due to the increasing cwnd exponentially in the slow start stage. During this stage, MSS-TCP increases its cwnd by a factor, as illustrated in Eq. (6) shows, forming a convex curve. However, when the sender receives 3DACKs, MSS-TCP reduces its cwnd by β as Algorithm 1 shows, and in the worst case, if the Retransmission Time-Out (RTO) is detected at any stage, MSS-TCP resets its cwnd to the initial value and triggers a slow start stage again.

3. Experimental design

This study conducted simulation experiments using the well-established network simulator ns-3 to assess the proposed CCAs. The

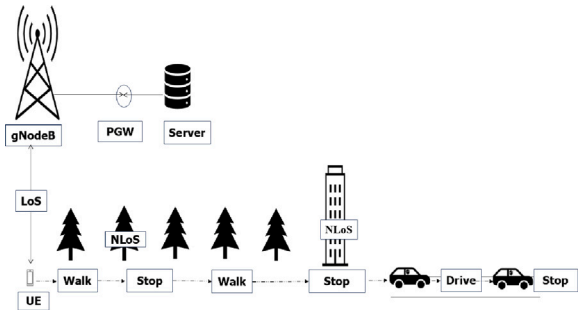


Fig. 2. Deployment scenario.

Table 1
Simulation parameters.

Attribute	Value
Carrier frequency	28 GHz
Loss model	Building Obstacle Propagation Loss Model
Bandwidth	1 GHz
TCP protocols	NewReno, HighSpeed, CUBIC, BBRV1, FB-TCP, MSS-TCP
Buffer size	0.25, 2.5, 20 MB
Initial RTO	1 s
PER	0, 10^{-9}, 10^{-8}, and 10^{-7}
Simulation time	60 s
Operating system	Ubuntu 20.04
Network simulation program	ns-3

ns-3 mmWave module, built on ns-3, encompasses key 3 GPP features, including channel modeling, dual connectivity support, and more. Therefore, ns-3 is an appropriate simulation tool for assessing 5G mmWave networks.

The simulated scenario replicates a cellular network in an urban area, incorporating UE and gNB stations under various conditions such as LoS and NLoS by implementing obstacles, like walking, driving, stopping, and moving. The MSS-TCP and state-of-the-art CCAs were evaluated in a scenario illustrated in Fig. 2. In this setup, a user initiates the experiment using a UE, initially pausing and then walking while maintaining a LoS connection with a gNB station. As UE progresses, it encounters trees, which serve as small obstacles to mimic NLoS conditions. To simulate a static NLoS scenario, the user pauses for four seconds behind a tree. Subsequently, the UE resumes movement, re-establishing LoS with the gNB station until reaching a building that simulates a large obstacle. The UE then pauses for four seconds behind the building, creating NLoS conditions. Finally, the UE moves towards a car, drives away from the gNB station, and stops after a certain distance.

In the scenario, the performance of the analyzed TCP CCAs is evaluated by applying different buffer sizes, specifically 0.25 MB, 2.5 MB, and 20 MB, representing small, BDP, and unlimited buffer sizes, respectively. Additionally, four PERs are applied to assess their impact alongside each buffer size on the performance of TCP variants. Table 1 shows a detailed overview of simulation parameters. Meanwhile, the simulation time for all experiments was set to 60 s, sufficient for TCP to reach its steady state interval to evaluate the CCA's performance optimally and ensure result accuracy [2,3]. Additionally, the average values of Key Performance Indicators (KPIs), including congestion window fluctuations, throughput, and latency, are calculated for each parameter set.

4. Results and discussion

This section evaluates MSS-TCP's behavior in comparison to benchmark TCP protocols. It also highlights performance outcomes related to congestion window fluctuations, throughput, and latency, illustrating the impact of buffer size and packet error rate on the proposed protocol's overall performance in cellular networks.

4.1. Congestion window (cwnd)

MSS-TCP effectively manages its sending rate, facilitating increased throughput and efficient recovery from losses, as demonstrated in Fig. 3, which highlights MSS-TCP's superior performance. The figure illustrates the protocol's adaptive response to various buffer and PER situations, adjusting the sending rate based on network feedback received through RTT parameter values. This adaptive approach aids in achieving higher throughput, mitigating the latency, and preventing buffer overflows. Despite achieving higher throughputs, MSS-TCP's mechanism enables it to tolerate buffer and PER loss. In contrast to other TCPs, MSS-TCP calculates RTT parameters reflecting the network's status and uses these parameters to decide on sending rate adjustments. This adjustment is achieved by modifying the increment of cwnd using the novel factor during the CA phase. Figs. 3(c) and 3(d) display MSS-TCP's cwnd adjustment under small and no PER loss, illustrating cwnd adjustment according to the buffer size applied and showcasing MSS-TCP's ability to react appropriately. Furthermore, MSS-TCP's adaptive behavior is evident in Fig. 3(b), where it reduces sending rates during NLoS conditions, swiftly recovers afterward, and achieves high sending rates in the CA phase. Additionally, in Fig. 3(a), cwnd adjustment for MSS-TCP is depicted when a high PER loss occurs in the network. This illustrates MSS-TCP's immunity to packet drops due to its CA phase mechanism. MSS-TCP estimates the upper bound of the network and updates cwnd based on the current RTT with MSS. This mechanism enables the protocol to operate around the maximum sending rate compared with other TCP variants, precisely adjusting its cwnd size in different buffer and PER conditions.

4.2. Average throughput

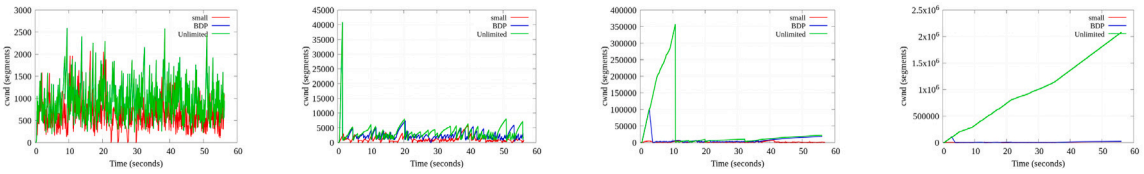
Fig. 4 presents the throughput results of the analyzed TCP algorithms across different buffer sizes and PERs within the examined network. MSS-TCP consistently outperforms other CCAs under most conditions in average throughput as illustrated in subfigures of Fig. 4, a success primarily attributed to its rapid cwnd growth facilitated by the novel factor mechanism. Moreover, MSS-TCP demonstrates lower sensitivity to small buffers and PER compared to loss-based TCP variants such as NewReno, HighSpeed, and CUBIC. These alternatives suffer significant performance deterioration with increasing PER due to their mechanisms for buffer filling, especially in NLoS conditions. This often leads to network congestion and subsequent packet drops.

MSS-TCP exhibits superior performance across various conditions, consistently outperforming loss-based TCP, BBR, and FB-TCP. Its ability to adapt and maintain efficient throughput in different buffer and PER scenarios, along with smoother transitions between LoS and NLoS states, highlights its robustness and effectiveness in diverse network environments.

4.3. Average latency

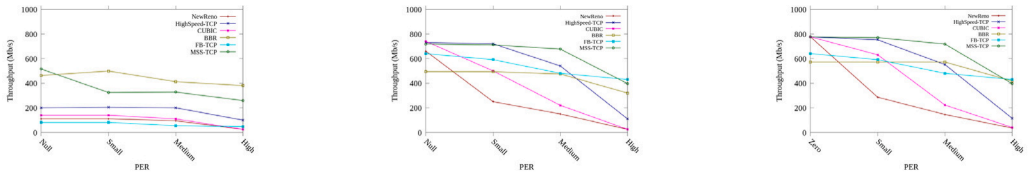
Fig. 5 depicts the latency results of TCP variants under various buffer sizes and PERs in a deployment scenario. MSS-TCP consistently maintains average latency with slight variations when compared to the latency of other TCP variants in most buffer sizes and PER scenarios, as depicted in subfigures Figs. 5(a), 5(b), and 5(c), except for scenarios with no PER. This performance is primarily attributed to MSS-TCP's increment factor mechanism. Additionally, MSS-TCP achieves lower latency than BBR in the unlimited case of buffer size when high PER is applied in the network, as illustrated in Fig. 5(c). The marginal latency difference observed with MSS-TCP is acceptable, considering its considerable improvements in throughput across various buffer sizes and PER scenarios.

Overall, these latency results highlight MSS-TCP's capability to maintain competitive latency performance while delivering significant throughput enhancements, positioning it as a promising choice for network deployments requiring both low latency and high throughput.



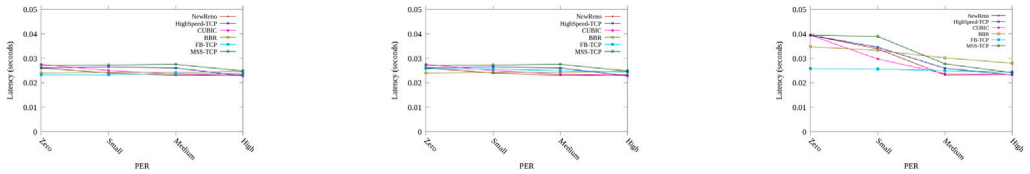
(a) High PER (b) Medium PER (c) Small PER (d) Null PER

Fig. 3. MSS-CCA cwnd across different buffer sizes and packet error rates.



(a) Small Buffer (0.25MB) (b) BDP Buffer (2.5MB) (c) Unlimited Buffer (20MB)

Fig. 4. Average throughput across different buffer sizes and packet error rates.



(a) Small Buffer (0.25MB) (b) BDP Buffer (2.5MB) (c) Unlimited Buffer (20MB)

Fig. 5. Average latency across different buffer sizes and packet error rates.

5. Conclusion

In this paper, a novel CCA named MSS-TCP has been introduced and evaluated. The primary innovation of this proposed algorithm lies in the incorporation of the increment factor mechanism, specifically tailored to MSS and RTT variations. The need for MSS-TCP arises from the limitations of existing CCAs in achieving optimal performance over 5G mmWave cellular networks, particularly under small buffer regimes and varying PER. Subsequently, a new CCA module was integrated into the ns-3 network simulator to evaluate its performance, comparing it to baseline CCAs in 5G mmWave cellular networks. Intensive simulation experiments assessed the performance of MSS-TCP against NewReno, HighSpeed, CUBIC, BBR, and FB-TCP. The results reveal that MSS-TCP achieves higher throughput than existing congestion control algorithms, while maintaining low latency and exhibiting reduced sensitivity to variations in buffer size and packet error rate. Moreover, MSS-TCP achieved significant throughput improvement, outperforming loss-based TCP algorithms, and by 24.26% to 45.43% in throughput compared to BBR when the buffer size matches the BDP size. In addition, there was a slight improvement in delay performance, reaching 2%–19% in some cases. Importantly, MSS-TCP, which functions as a sender-side TCP module, introduces no changes to the receiver-side or network routers, utilizing and featuring a novel CCA driven by the factor mechanism. Our future work aims to extend the evaluation of MSS-TCP through live experiments, measuring the fairness between MSS-TCP and other TCP variants to assess equitable link sharing. In addition, we plan to evaluate the proposed protocol across various networks.

CRediT authorship contribution statement

Omar Imhemed Alramli: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Conceptualization. **Zurina Mohd Hanapi:** Writing – review & editing, Supervision. **Mohamed Othman:** Supervision. **Normalia Samian:** Supervision. **Idawaty Ahmad:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The authors would like to acknowledge the support from the Research Management Centre (RMC) of Universiti Putra Malaysia, Malaysia.

References

[1] E. Khorov, A. Krasilov, M. Susloparov, L. Kong, Boosting TCP & QUIC performance in mmWave, terahertz, and lightwave wireless networks: A survey, IEEE Commun. Surv. Tutor. (2023) http://dx.doi.org/10.1109/COMST.2023.3301820.

[2] R. Poorzare, A.C. Augé, How sufficient is TCP when deployed in 5G mmWave networks over the urban deployment? IEEE Access 9 (2021) 36342–36355, http://dx.doi.org/10.1109/ACCESS.2021.3063623.

[3] M. Zhang, M. Polese, M. Mezzavilla, J. Zhu, S. Rangan, S. Panwar, M. Zorzi, Will TCP work in mmWave 5G cellular networks? IEEE Commun. Mag. 57 (1) (2019) 65–71, http://dx.doi.org/10.1109/MCOM.2018.1701370.

[4] M. Polese, R. Jana, M. Zorzi, TCP in 5G mmWave networks: Link level retransmissions and MP-TCP, in: 2017 IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS, IEEE, 2017, pp. 343–348, http://dx.doi.org/10.1109/INFCOMW.2017.8116400.

[5] G.-H. Kim, Y.-Z. Cho, mmS-TCP: Scalable TCP for improving throughput and fairness in 5G mmWave networks, Sensors 22 (10) (2022) 3609, http://dx.doi.org/10.3390/s22103609.

[6] Y. Ren, W. Yang, X. Zhou, H. Chen, B. Liu, A survey on TCP over mmWave, Comput. Commun. 171 (2021) 80–88, http://dx.doi.org/10.1016/j.comcom.2021.01.032.

[7] A. Afanasyev, N. Tilley, P. Reiher, L. Kleinrock, Host-to-host congestion control for TCP, IEEE Commun. Surv. Tutor. 12 (3) (2010) 304–342, http://dx.doi.org/10.1109/SURV.2010.042710.00114.

[8] V.K. Sharma, L.P. Verma, M. Kumar, CL-ADSP: Cross-Layer adaptive data scheduling policy in mobile ad-hoc networks, Future Gener. Comput. Syst. 97 (2019) 530–563.

[9] V.K. Sharma, M. Kumar, Adaptive congestion control scheme in mobile ad-hoc networks, Peer-to-Peer Netw. Appl. 10 (2017) 633–657.
[10] J. Lorincz, Z. Klarin, J. Ožegović, A comprehensive overview of TCP congestion control in 5G networks: Research challenges and future perspectives, Sensors 21 (13) (2021) 4510, http://dx.doi.org/10.3390/s21134510.
[11] R. Poorzare, A.C. Augé, FB-TCP: A 5G mmWave friendly TCP for urban deployments, IEEE Access 9 (2021) 82812–82832, http://dx.doi.org/10.1109/ACCESS.2021.3087239.
[12] H. Haile, K.-J. Grinnemo, S. Ferlin, P. Hurtig, A. Brunstrom, End-to-end congestion control approaches for high throughput and low delay in 4G/5G cellular networks, Comput. Netw. 186 (2021) 107692, http://dx.doi.org/10.1016/j.comnet.2020.107692.
[13] L.P. Verma, V.K. Sharma, M. Kumar, D. Kanellopoulos, A novel delay-based adaptive congestion control TCP variant, Comput. Electr. Eng. 101 (2022) 108076, http://dx.doi.org/10.1016/j.compeleceng.2022.108076.
[14] M.R. Kanagarathinam, S. Singh, I. Sandeep, H. Kim, M.K. Maheshwari, J. Hwang, A. Roy, N. Saxena, NexGen D-TCP: Next generation dynamic TCP congestion control algorithm, IEEE Access 8 (2020) 164482–164496, http://dx.doi.org/10.1109/ACCESS.2020.3022284.
[15] Y. Xie, X. Jiang, G. Gong, Z. Jiang, G. Jin, H. Chen, Yinker: A flexible BBR to achieve the high-throughput and low-latency data transmission over Wi-Fi and 5G networks, Comput. Netw. 222 (2023) 109530, http://dx.doi.org/10.1016/j.comnet.2022.109530.
[16] L. Guo, J.Y. Lee, TCP-FLASH-A fast reacting TCP for modern networks, IEEE Access 9 (2021) 68861–68879, http://dx.doi.org/10.1109/ACCESS.2021.3077612.
[17] H. Haile, K.-J. Grinnemo, P. Hurtig, A. Brunstrom, RBBR: A receiver-driven BBR in quic for low-latency in cellular networks, IEEE Access 10 (2022) 18707–18719, http://dx.doi.org/10.1109/ACCESS.2022.3148998.
[18] T. Henderson, S. Floyd, A. Gurtov, Y. Nishida, The NewReno modification to TCP's fast recovery algorithm, 2012, http://dx.doi.org/10.17487/RFC3782.
[19] S. Floyd, HighSpeed TCP for large congestion windows, 2003, http://dx.doi.org/10.17487/RFC3649.
[20] S. Ha, I. Rhee, L. Xu, CUBIC: a new TCP-friendly high-speed TCP variant, ACM SIGOPS Oper. Syst. Rev. 42 (5) (2008) 64–74, http://dx.doi.org/10.1145/1400097.1400105.
[21] N. Cardwell, Y. Cheng, C.S. Gunn, S.H. Yeganeh, V. Jacobson, BBR: Congestion-based congestion control, Commun. ACM 60 (2) (2017) 58–66, http://dx.doi.org/10.1145/3009824.
[22] O.I. Alramli, Z.M. Hanapi, M. Othman, I. Ahmad, N. Samian, RTTV-TCP: Adaptive congestion control algorithm based on RTT variations for mmWave networks, Ad Hoc Netw. 164 (2024) 103611, http://dx.doi.org/10.1016/j.adhoc.2024.103611.
[23] Z. Wang, J. Crowcroft, Eliminating periodic packet losses in the 4.3-Tahoe BSD TCP congestion control algorithm, ACM SIGCOMM Comput. Commun. Rev. 22 (2) (1992) 9–16, http://dx.doi.org/10.1145/141800.141801.
[24] L.S. Brakmo, L.L. Peterson, TCP Vegas: End to end congestion avoidance on a global Internet, IEEE J. Sel. Areas Commun. 13 (8) (1995) 1465–1480, http://dx.doi.org/10.1109/49.464716.
[25] M.A. Alrshah, M. Othman, B. Ali, Z.M. Hanapi, Comparative study of high-speed Linux TCP variants over high-BDP networks, J. Netw. Comput. Appl. 43 (2014) 66–75.
[26] C. Jin, D. Wei, S.H. Low, J. Bunn, H.D. Choe, J.C. Doylle, H. Newman, S. Ravot, S. Singh, F. Paganini, et al., FAST TCP: From theory to experiments, IEEE Netw. 19 (1) (2005) 4–11, http://dx.doi.org/10.1109/MNET.2005.1383434.
[27] W. Pan, Y. Xu, S. Liu, Making TCP BBR pacing adaptive with domain knowledge assisted reinforcement learning, IEEE Trans. Netw. Sci. Eng. (2023) http://dx.doi.org/10.1145/141800.141801.
[28] V. Cerf, R. Kahn, A protocol for packet network intercommunication, IEEE Trans. Commun. 22 (5) (1974) 637–648, http://dx.doi.org/10.1109/TCOM.1974.1092259.
[29] M. Allman, S. Floyd, C. Partridge, RFC3390: Increasing TCP's initial window, 2002, Available online: https://www.rfc-editor.org/rfc/rfc3782 (accessed 10 June 2024).
[30] J. Chu, N. Dukkipati, Y. Cheng, M. Mathis, RFC 6928: Increasing TCP's initial window, 2013, Available online: https://datatracker.ietf.org/doc/html/rfc6928 (accessed 25 May 2024).