Journal of Network and Systems Management (2022) 30: 67
https://doi.org/10.1007/s10922-022-09677-1

DB-CMT: A New Concurrent Multi-path Stream Control Transport Protocol

Lal Pratap Verma[1] · **Varun Kumar Sharma**[2] · **Mahesh Kumar**[3] · **Dimitris Kanellopoulos**[4] · **Aniket Mahanti**[5]

Received: 10 June 2021 / Revised: 14 June 2022 / Accepted: 3 July 2022 /

Abstract

Stream Control Transmission Protocol (SCTP) exploits multiple network interfaces to provide multi-streaming and data chunk ordering in a stream. An extended feature of SCTP, i.e., Concurrent Multi-path Transfer (CMT), bids concurrent data transmission in a multi-path data transfer environment and guarantees bandwidth aggregation, load sharing, robustness, and reliability. In such an environment, the paths usually have distinct characteristics (i.e., delay, Packet Loss Rate (PLR), and bandwidth). Thus, data chunks are received out-of-ordered at the destination. As a result, CMT causes excessive receiver buffer blocking and unnecessary congestion window (*cwnd*) reductions. Also, during the selection of the retransmission destination path (to resend a lost data chunk), CMT does not take into account vital Quality of Service (QoS) parameters such as the PLR of a path under consideration. This paper introduces a new Delay-Based Concurrent Multi-path Transfer (DB-CMT) approach that transmits data on multiple paths according to their delay. In this scheme, we present a Delay-Based Data chunk Scheduling Policy (DB-DSP), a Retransmission Path Selection Policy (RTX-CL), and a new Delay-Based Fast Retransmission Policy (DB-FRP). The simulation results show that the DB-CMT's RTX-CL policy performs better than the well-known RTX-CWND and RTX-LOSSRATE retransmission schemes. Also, the overall performance of DB-CMT witnesses improved throughput, fewer timeouts, and reduced File Transfer Time (FTT) performances.

Keywords Congestion window · Multi-path · SCTP · CMT · RTX · Path quality

---

✉ Aniket Mahanti
a.mahanti@auckland.ac.nz

Extended author information available on the last page of the article

1 Introduction

The traditional Transmission Control Protocol/Internet Protocol (TCP/IP) model transmits data packets by selecting only a unique ‘best available network path’ [1]. This model and its classical Transport Layer (L-4) protocols (i.e., Transmission Control Protocol (TCP) [2] and User Datagram Protocol (UDP) [3]) adopt only the single-path communication paradigm. However, they cannot satisfy the fault tolerance demands and QoS requirements imposed by multimedia applications over the Internet. Later, the multi-path communication paradigm-based protocols overcame these insufficiencies of the single-path communication paradigm-based protocols. Multi-pathing activates inverse multiplexing of network resources to send data packets over a set of available network paths instead of a single path. Numerous research efforts from academia, industry, and international standards organizations (e.g., IEEE and IETF) recently adopted the multi-path concept. For example, the research community proposed a plethora of L-4-based policies that rely upon concurrent transmission features. Multi-pathing is also necessary as the two end-hosts must sustain more than one path to guarantee the resiliency and consistency of the network [4–9].

The transmission of real-time video requires low End-to-End (EtE) delay and high bandwidth. These restrictions can deteriorate the synchronization between voice and video, leading to poor video quality. Meanwhile, the continuous success of video streaming services (e.g., YouTube) depends on delivering high video quality to the end-users [10, 11]. In such an environment, online real-time video applications (e.g., video-conferencing) demand effective resource management policies that adopt the multi-path concept. Existing wireless technologies and standards (e.g., Long-Term Evolution (LTE), WiFi, and WiMax) and the increasing storage and computational capabilities of new smartphones encourage the mandate for multimedia services. New wireless portable devices have multiple network interfaces to provide smooth and high-quality service provisioning in a heterogeneous wireless environment. Multi-homed devices can increase their throughput using parallel transmissions over numerous paths and bandwidth aggregation. For instance, a good smartphone can transmit data simultaneously if equipped with WiFi and 3G/4G/5G interfaces. Data transmission can occur in the path with the highest throughput, while the backup process can happen in the slower paths. Therefore, multi-homing protocols can cover the fault tolerance demands of multimedia applications [12]. It is noteworthy that single-path L-4 protocols do not support multi-homing and bandwidth aggregation. The multi-homing feature offers two end-user systems to launch a logical association over the available network interfaces. This logical association allows the SCTP sender to convey data to the multi-homed receiver through multiple paths. Initially, SCTP uses a path with the highest throughput as a major path and starts data transmission. The remaining available paths are used for data transmission when the primary path becomes unreachable due to network congestion or connection failure. In case of a path failure, the multi-homing feature of SCTP provides a backup path.

The QoS requirements of multimedia applications are satisfied if the available network resources are appropriately utilized [13–17]. SCTP [18], CMT [19], and Multi-path TCP (MPTCP) [20–22] are L-4 protocols that aim to provide appropriate stability between strict QoS requirements and utilize available network resources efficiently. SCTP and MPTCP take advantage of multiple network interfaces and support multi-homing. Concerning fault tolerance, they also provide elective reliability. During the transmission association establishment amid two end-hosts, SCTP and CMT exploit multi-homing. In an already established connection, MPTCP contrasts SCTP by systemizing several TCP subflows simultaneously. MPTCP integration has been done with Apple's iOS version 7.0. MPTCP smartphones can use the Gigapath commercial service to achieve around 800 Mbps throughput by joining WiFi and LTE networks [23, 24]. Besides, OVH Telecom and Tessares introduced some bandwidth aggregation-based multi-path schemes. OVH Telecom introduced a new product called 'OverTheBox' [25], which integrates SOCKS proxies and MPTCP to facilitate users to combine numerous Digital Subscriber Lines (DSLs). Tessares introduced a network service (on top of MPTCP) that performs bandwidth aggregation of multiple network infrastructures (i.e., DSL or LTE) [1, 26].

2 Motivation and Scope of the Article

SCTP initially does not support the notion of simultaneous multi-path data transfer. It motivated Iyengar et al. [19] to introduce the idea of CMT as an extension of SCTP. CMT competently offers SCTP load sharing, fault tolerance, and bandwidth aggregation capabilities [27–29]. However, this idea works well for paths having symmetric characteristics (i.e., similar PLR, delays, and bandwidth). Asymmetric path characteristics, which are highly likely for modern Internet arrangements involving numerous service providers, make things extremely challenging for CMT. Each path may have different characteristics, and because of this, it may take different times for the data to reach the destination. As a result, the destination frequently receives unordered data. However, CMT uses Transmission Sequence Numbers (TSNs) and a limited size buffer at the receiver to reorder the data chunk. And the receiver cannot erase the data chunk from the receiver buffer until it receives the ordered data chunk. The continuous unordered data chunk reception leads to the receiver buffer blocking problem [15, 19, 30] that seriously degrades the performance at the application level. The other issue with CMT is that it assumes all paths are disjointed. Hence, the CMT cannot acclimatize its functionality when many bottleneck links get involved in the path. Moreover, the uncoupled congestion control scheme has been implemented on top of CMT. Hence, likewise TCPReno, CMT performs uncoupled (independent) congestion control on each path; therefore, it cannot attain flexible load-balancing, ultimately hampering its fairness and throughput performances. Another drawback of CMT is that it uses a Round Robin (RR)-based data scheduling notion to distribute data over multiple available network paths. This notion does not consider a path's fluctuating characteristics while scheduling data chunks. It causes the problem of unordered data chunk delivery at the destination, leading to the receiver buffer blocking issue. Due to the continuous unordered data chunk reception, the CMT receiver sends an instant Selective ACKnowledgment (SACK) to the sender. Consequently, the sender needlessly has to reduce the *cwnd*, whereas the network is not congested. Another problem relates to the policies for selecting the retransmission path that resends the lost data chunk. The recommended path selection policies RTX-CWND [19] and RTX-SSTHRESH [19] choose the path that has the largest *cwnd* or slow-start threshold (ssthresh), respectively. However, both approaches choose this path randomly in case of an exceptional condition. For instance, a particular situation appears when multiple paths have identical *cwnd* or ssthresh values. Such random selection may incorrectly choose the path having the maximum PLR and minimum bandwidth and may cause different problems. We conclude that new efficient path selection schemes and retransmission methods are required from the above discussion.

This paper introduces a new DB-CMT scheme and contributes as follows:

- DB-CMT transmits data chunks over numerous network interfaces (underlying paths) according to their respective traffic load. In DB-CMT, the path load is estimated through delay and *cwnd* parameters. This new scheduling policy is called DB-DSP.
- The DB-CMT scheme incorporates a novel retransmission destination selection strategy (RTX-CL) to avoid the random selection of retransmission destinations.
- The DB-CMT scheme has a new fast retransmission method to reduce the needless *cwnd* reductions due to unordered data chunk delivery. This original method is called DB-FRP.
- Confirmation of the effectiveness of DB-CMT by conducting extensive experiments and comprehensive evaluation. The validation of DB-CMT has been carried out on network simulator (*ns*)-2.
- The simulation results exhibit the efficacy of DB-CMT and show how DB-CMT outperforms other CMT approaches in terms of throughput and FTT performances.

The arrangement of the paper is as follows. Section 3 presents the related work of CMT and MPTCP-Based policies. Section 4 introduces the proposed DB-CMT scheme. Section 5 presents the simulation setup, comprehensive performance evaluation of considered the CMT schemes. Section 6 presents the summary of assessed simulation results. Section 7 finally concludes the paper.

3 Related Work

This section describes the significant proposals based on multi-path L-4 protocols and the key issues that affect their performance. We have divided this section into two parts which are as follows: (I). SCTP-Based CMT Schemes, and (II). MPTCP-Based Schemes.

3.1 SCTP-Based CMT Schemes

Each path may have different QoS factors like bandwidth and delay. Due to dissimilarity in such characteristics, CMT severely suffers from the issue of receiver buffer blocking, unsolicited *cwnd* reductions, needless retransmissions, and inaccurate data scheduling [1, 4, 27–29, 31, 32]. Iyengar et al. [19] analyzed and indicated that the CMT paradigm suffers from the issue of unnecessary retransmissions. Therefore, considering this issue, the authors suggested a solution known as Split Fast Retransmit (SFR). SFR enhanced the conventional fast retransmission scheme by hosting the concept of a virtual queue per destination. This concept assists SFR in inferring the missing segments more precisely. Moreover, they also identified the *cwnd* update problem and advocated a *cwnd* Update scheme for CMT (CUC). CUC handles the after-effects of abridged *cwnd* growth owing to lesser *cwnd* updates. From another perspective, Dreibholz et al. [33] recommended the Sender Buffer Splitting (SBS) method, which splits the sender buffer conferring to the existing paths. SBS handles the blocking problem when the receiver blocks data chunks' elimination due to out-of-order delivery. However, SBS suffers from local buffer blocking due to each path's different delay and bandwidth values. Dreibholz et al. [34] presented a report on SCTP past, present, and future standardization and identified the activities and challenges in the CMT paradigm. Wallace and Shami [27] highlighted various problems related to the CMT scheme, such as unnecessary fast retransmissions, excessive network traffic, receiver buffer blocking, crippled *cwnd* growth, and naive scheduling. Verma et al. [28] estimated and utilized the path delay factor to adjust the *cwnd* size and ssthresh value. In this policy, the adaptations in *cwnd* depend upon packet loss and out-of-order data chunk delivery events. This policy performs minor and significant reductions in *cwnd* size during out-of-order data chunk delivery and packet loss events. Yang et al. [30] advocated a modified fast retransmission scheme that utilizes both loss rate and delays to reduce the possibility of receiver buffer blocking. These policies [28, 30] use conventional CMT's independent congestion control scheme. Hence, it is necessary to evaluate these policies [28, 30] on the fairness parameter because they might be suffering from the issue of being less fair towards other competing TCP flows. Natarajan et al. [35] suggested a new state, called Potentially Failed (PF) with CMT (CMT-PF), to diminish the receiver buffer blocking that happens due to path failure. This state shows whether the destination is reachable or not. Thus, all the new data packets are forwarded to another alternative path. Yilmaz et al. [36] proposed a Non-Renegable SACK (NR-SACK) policy. This policy eliminates the segment from the receiver buffer without caring for *cwnd* growth and reordering.

Further, in a heterogeneous wireless network environment, data are sent over heterogeneous paths, and these paths may have different characteristics and QoS parameters. Xu et al. [37] suggested a Quality-Aware adaptive multi-path data transfer policy (CMT-QA) for a distinct network (wireless) environment to schedule data over the numerous paths rendering available path superiority. Xu et al. [38] further enhanced CMT considering the same environment and suggested a Network Coding grounded CMT (CMT-NC). Singh et al. [31] surveyed existing multi-path routing and traffic splitting approaches. They also discussed the problems and challenges of inter-layer cooperation, scalability, stability, buffering, and packet reordering in multi-path provisioning. As a solution to data packet reordering and crippled *cwnd* growth, Shailendra et al. [39] recommended Multi-path SCTP (MPSCTP). Later, Shailendra et al. [40] appraised MPSCTP to regulate the data transfer on an individual path, rendering the total EtE delay. This method minimizes the average packet delay over multiple available network paths but suffers from lower resource utilization complications due to its identical data dissemination strategy. Shailendra et al. [41] also advocated a Tx-CWND retransmission destination selection method to increase the performance of MPSCTP. Meanwhile, Hwang et al. [42] suggested a network coding-based scheme that deals with the buffer blocking problem of CMT. This scheme utilizes Luby transform codes to minimize the computational overhead and retransmission of data. The scheme's main objective is to reduce the necessity for retransmissions and in-order delivery. However, there is a reasonable chance that this scheme may suffer from higher per-packet in order delivery delay. It may be because this policy incorporates the usage of block codes and the delay generally increases with the block code size. Verma et al. [43] proposed a delay-based packet scheduling approach that uses the path load variation factor and different threshold variables to schedule the data on multiple paths.

Recently, the application of CMT has been widely recommended (see [44–52] and References therein) from the perspective of real-time video traffic transmission. Initially, Wu et al. [44] gave Distortion-Aware CMT (CMT-DA) scheme concerning a heterogeneous network environment. CMT-DA suggests introducing the EtE distortion video model on top of CMT. This scheme minimizes video distortion at the flow level by decreasing the effective PLR. Moreover, in the perspective of high-quality video delivery over critical channel conditions, it is necessary to assess the content factors (i.e., decoding dependency amongst frames (video) and priorities of frame), and the scheme should accomplish segregated frame transmission. Nonetheless, CMT-DA alleviates video distortion without considering the content factors. Regarding this issue, Wu et al. [45] gave a Content-Aware CMT (CMT-CA) scheme considering a heterogeneous wireless environment. CMT-CA reduces video distortion by competently considering the aforementioned content factors and utilizing the limited wireless channel capacity. These schemes [44, 45] offer transmission consistency by relying on the conventional standard of retransmission. Further, Wu et al. [46] suggested a Video and Raptor code-conscious CMT (CMT-VR) scheme that considers the challenges in incorporating the CMT paradigm with real-time video streaming. CMT-VR utilizes the proposed raptor coding mechanism, which offers a frame-level guard for video data. CMT-VR also comprises an online chunk scheduling and retransmission scheme aiming to maximize the channel capacity by mitigating needless retransmissions. The CMT-VR method suggests transmission consistency by relying on the mechanism of raptor codes and a modified retransmission scheme. Typically, the schemes [44–46] offer transmission reliability either by traditional or flexible retransmission policies. Undeniably, these schemes suffer from low QoS while dealing with real-time video applications that are highly delay-sensitive. Further, Chen et al. [47] recommended a strict requirement of realizing the inter-packet dependency that previous approaches [44–46] do not consider. Generally, in mobile (wireless) broadcast networks, the transmission policies are designed to assist the worst-case user. Whenever any transmission policy deals with the scheduling of real-time video application packets, retransmission is a significant concern before that policy. When considering the critical network environment, a packet can be lost due to many reasons like buffer-overflow and transmission errors. When we consider the case of packet loss due to transmission errors, the previously suggested transmission policies utilize Forwarding Error Correction (FEC) mechanism to detect and correct erroneous symbols. For this, the policies transmit redundant symbols in the form of complementary restoration symbols. These restoration symbols let the receivers restructure the original symbols even if some are not received correctly due to errors in transmission [48]. Further, the authors [47] show their concerns about using traditional fountain codes in the abovementioned approaches. Since such codes cannot offer flexibility in guarding the source-generated symbols, these codes usually have the limited ability to provide an alike type of protection for all source-generated symbols without considering the inter-packet dependency. It might lead to the issue that some original/received or reconstructed symbols may become unusable for the final video restoration when their interlinked references get lost or unrecoverable. Hence, the authors [47] modify fountain codes' conventional equal protection policy by leveraging the inter-packet dependency in the suggested model. Recently, Chen et al. [49] showed concerns about a strict long-term rate control mechanism requirement for concurrent multi-path video transmission service. Since the policies [44, 50] do not consider the quality measure of long-term video transmission, the buffer usage of the current Intra (I) video frame affects the quality of the next Predicted (P) video frames. It motivated the authors [49] to introduce a novel policy of a long-term rate control mechanism that assists a concurrent multi-path video transmission service.

3.2 MPTCP-Based Schemes

An additional connection-oriented protocol that facilitates multi-homing is MPTCP. The performance of MPTCP is better with the incorporation of middle-boxes in contemporary Internet architecture [4, 20]. Yedugundla et al. [53] studied the performance of CMT and MPTCP for delay subtle traffic and observed that both methods decrease the network delay suggestively in a symmetric packet loss environment. Nevertheless, transmission delay minimization is not up to the mark in an asymmetric delay and loss network environment. However, the applications may still take benefit from an additional feature (i.e., fault tolerance) of multi-path communication without increasing the delay. Multi-streaming is another notion initially utilized in SCTP to accomplish varied network circumstances. MPTCP inherits this feature from SCTP, where Li et al. [54] introduced a Network Coding-based MPTCP (NC-MPTCP) to circumvent retransmission in case of delay. However, Systematic Coding MPTCP (SC-MPTCP) [55] practices redundant code to diminish the needless fast retransmission. Cui et al. [56] advocated an alternative augmented version of MPTCP based on foundation codes. In contrast, Zhou et al. [57] presented the *CWnd* Adaptation MPTCP (CWA-MPTCP) scheme, suggesting the concept of dynamic adaptations in *cwnd* of each subflow following its changing EtE path delay. Xu et al.

[15] proposed a Cross-Layer Fairness-Driven CMT (CMT-CL/FD) scheme for video transmission over the heterogeneous wireless network to diminish reordering during the transmissions.

In the past, a significant amount of work has been done concentrating on MPTCP scheduler design. It is the responsibility of a scheduler to distribute data chunks over multiple interfaces (paths). And it is certain that the performance of such multi-path protocols largely depends on their scheduler design. This is because the imprecise scheduling decisions might lead to Head-of-Line (HoL) blocking, out-of-order delivery, and receiver buffer blocking issues, especially when the paths are asymmetric. Undeniably, in such a case, the users perceive low throughput performance as well as high delays, resulting in inferior Quality of Experience (QoE) for the user [58, 59]. Initially, the conventional scheduler in MPTCP utilizes the Round Trip Time (RTT) estimations. This scheduler chooses to transmit over the fast path (i.e., minimum RTT delay path). It prefers to fill the *cwnd* of the subflow with the minimum RTT before selecting the other subflows with higher RTTs. Now, when any of these scheduled subflows blocks the created connection, this scheduler adapts itself by retransmitting those chunks (i.e., which creates blocking) on the fast path and penalizing the slow path (i.e., higher RTT delay path), which causes that blocking issue. Moreover, this causes non-optimality in bandwidth aggregation since slower paths are underutilized [60–63]. Further, this scenario worsens when the paths are heterogeneous, where RTTs and the available bandwidth of the paths vary substantially. This further leads to the severe issue of receiver buffer blocking due to high out-of-order deliveries, which ultimately reduces the QoE to the user. Recently, some new schedulers have also been suggested, such as Earliest Completion First (ECF) [64], BLock ESTimation (BLEST) [60], Slide Together Multi-path Scheduler (STMS) [65], and Peekaboo [66]. In particular, to prevent non-optimal bandwidth aggregation problems, ECF includes the bandwidth estimation and the amount of data queued in the sender buffer factors along with MPTCP's default scheduler. By identifying whether utilizing a slower path for transmitted chunks will cause faster paths to be underutilized, ECF uses the faster path more competently and reduces the receiver buffer blocking. ECF and BLEST schedulers address about HoL blocking issue by suggesting a wait state. By introducing such a state, the scheduler can determine when to halt for more stable conditions to come for scheduling data chunks. Nevertheless, stopping and looking for more stable conditions would cause underutilization, especially when the path properties vary dynamically. Hence, Wu et al. [66] motivated and suggested a new scheduler called Peekaboo that considers the dynamic, varying path properties while transmitting data chunks. This scheme adapts the data chunk scheduling criteria by observing the effects caused by the continuously varying dynamicity level of available network paths. This scheme works in two folds: Firstly, Peekaboo chooses a deterministic approach to deal with varying dynamicity levels of the path by utilizing a self-learning adaptive and BLEST's halt approach. Secondly, Peekaboo introduces a stochastic modification policy to better handle non-optimal deterministic scheduling decisions. Shi et al. [65] identified that in a constrained buffer environment at the host, the aggregated throughput performance is far limited compared with two single-path TCP connections combined under similar network and buffer configurations. Also, they have acknowledged that the state-of-the-art schedulers require a large buffer on the fast path. So the authors gave a new scheduling scheme called STMS to handle both the issues well. STMS scheduler takes care that the fast path should continuously transmit packets with sequence numbers smaller than those of the slow paths. Hurtig et al. [67] highlighted the low-latency capacity aggregation issue by considering various conventional and new MPTCP scheduling policies. They demonstrated that various MPTCP schedulers could not offer appropriate stability and performance in asymmetric path conditions (primarily when the MPTCP scheduler deals with interactive applications). They cannot utilize the aggregate capacity and do not offer low latency as well. Two schedulers have been suggested to address the problem related to asymmetric characteristics possessed by the paths: Shortest Transfer Time First (STTF) [68] and BLEST scheduler. The STTF tries to reduce the transmission time of each segment, while the BLEST tries to lessen the buffer blocking problem. The STTF scheduler is multi-faceted and needs more resources. However, comparatively, BLEST is a lightweight scheduler. They have extensively designed and implemented these two schedulers mentioned above in the Linux Kernel environment and compared the performance with the default MPTCP scheduler. STTF estimates the probable transmission time of a chunk considering the path properties of all subflows. STTF effectively schedules all the un-transmitted chunks on the fastest existing subflow without caring about the current status of subflow's *cwnd*. STTF estimates the probable transmission time for a chunk over a specific subflow considering smoothen RTT (sRTT), traffic status of the subflow, and the amount of data queued in the sender buffer factors. Specifically, the traffic status of the subflow is one factor that differentiates STTF from the other state-of-the-art schedulers. Since most schedulers often blindly assume that the connections are always in the Congestion Avoidance (CA) phase, a hypothesis is often incorrect in the case of short-lived flows. These schedulers, such as BLEST, STMS, ECF, and STTF, have the shared motivation to reduce the buffer blocking problem and keep lower latency in asymmetric paths [69]. They ultimately realize the subflow, which has higher RTT. They schedule the packets to that subflow with higher sequence numbers. Accordingly, these schedulers manage to overcome the problem of out-of-order receptions. Further, a flavor of scheduler based on MPTCP considering ad-hoc network environment called Cross-Layer Adaptive Data Scheduling Policy (CL-ADSP) suggested by Sharma et al. [29]. CL-ADSP considers the paths' dissimilar properties while scheduling the data chunks. CL-ADSP has been designed and implemented using estimated RTT and average Medium Access Control (MAC) layer retries as a metric in an ad-hoc network environment on top MPTCP. However, when the RTT estimations are utilized as a network congestion metric, a lesser channel utilization may be attained in the existence of reverse traffic. Recently, Gao et al. [70] suggested a new packet-level-based load balancing scheme called Adaptively Adjusting Concurrency (AAC), considering high latency and packet reordering. AAC makes reasonable amendments in the number of chosen paths based on identifying differences in their properties. Table 1 summarizes SCTP (CMT) and MPTCP-Based schemes, while Table 2 presents their advantages and disadvantages.

Table 1 Summary of SCTP/CMT-based and MPTCP-based schemes

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
SCTP/CMT-based schemes								
CMT-VR [46] (2018)	L-4	GP, EtE-D, V-PSNR	InD	NS	PPB	NS	CMT-VR has addressed the challenges while incorporating the CMT feature with real-time video streaming	General
CMT-SR [74] (2018)	L-4	TP, SSIM, V-PSNR	InD	NS	NS	NS	CMT-SR has addressed the complex features of the existing CMT schemes, which hinders the possibility of their deployment for traditional video streaming services	Wired
Verma et al. [28] (2018), A-CMT [71] (2017)	L-4	TP	InD	UC	PPB	CnS	These schemes have addressed the issues of the CMT scheduler when the paths' characteristics vary substantially	Wired
Arianpoo et al. [75] (2017), CMT-NC [38] (2016)	L-4	TP, ADD, GP	InD	UC	PPB	CnS	These schemes have dealt with the receiver buffer blocking problem	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
CMT-CA [45] (2016)	L-4	GP, EtE-D, V-PSNR	InD	UC	PA	CnS	CMT-CA has addressed how conventional CMT variants have not optimally utilized limited wireless network resources when dealing with video streaming applications	General
CMT-CQA [76] (2015)	L-4, L-2 (CL)	TP	InD	UC	PA	CnS	CMT-CQA deals with the issue of receiver buffer blocking in the critical environment by jointly considering L-4 and MAC layer-based QoS parameters and path history information	Wireless
Optimized MPSCTP [41] (2015)	L-4	TP, FTT	InD	UC	PA	CnS	Optimized MSTCP: Tx-CWND path selection policy and heuristic approach to implementing DIO in MPSCTP	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
CMT-CL/FD [15] (2014)	L-4, L-2 (CL)	TP, FN, V-PSNR	InD	UC	NS	NS	CMT-CL/FD states the ineffectiveness of the current CMT schemes in handling dynamic wireless channel characteristics by addressing the issues of data scheduling and fairness	Wireless
CMT-DA [44] (2014)	L-4	TP, GP, IPD	InD	UC	PPB	CnS	CMT-DA deals with the problem of delivering good quality real-time video data with available CMT policies due to their stringent QoS requirements and different path characteristics in a heterogeneous network environment	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
Min–Max MPSCTP [40] (2013), MPSCTP [39] (2011)	L-4	FTT, TP, APDD	DP	UC	NS	NS	These schemes act as an augmented form for SCTP for CMT. Additionally, these schemes handle the issue of packet reordering and crippled *cwnd* growth during CMT	General
CMT-QA [37] (2012)	L-4	TP, ARD	InD	UC	PA	CnS	Path quality-based data distribution and retransmission	General
Adhari et al. [77] (2011)	L-4	PTP	SP	UC	PA	CnS	This scheme assessed the challenges of CMT over paths having different characteristics. It also deals with the CMT scheme's sender and receiver buffer blocking issues	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
Dreibholz et al. [33, 34] (2010) and (2011)	L-4	PTP	SP	UC	PA	NS	These schemes assessed the challenges of CMT over paths having distinct characteristics. Both methods deal with the sender and receiver buffer blocking issues of the CMT scheme	General
CMT-PF [35, 78] (2009) (2008)	L-4	TP, GP	InD	UC	PPB	CnS	CMT-PF scheme assesses and revises CMT retransmission and failure recognition policies. CMT-PF also handles the buffer blocking issue associated with CMT	General
W-PR-SCTP [79] (2007)	L-4	TP, J	DP	UC with SBD	NS	UB	Adaptive load balancing on multiple interfaces (underlying paths) with diverse characteristics	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
CMT [19] (2006)	L-4	TP, FTT	InD	UC with SBD	PPB	CnS	After-effects of reordering acquaint with CMT: 1. Needless FR procedure performed by the sender 2. Exceedingly conventional *cwnd* growth at the sender 3. Higher ACK traffic	General
W-SCTP [80] (2004)	L-4	ARD, ATT	DP	UC	PA(R) / PPB(S)	NS	Load balancing over multiple available network connections	General
LS-SCTP [81, 82] (2004)	L-4	TP	AP	UC	PA	CnS	Load balancing over multiple available network connection	Wireless
BA-SCTP [83] (2003)	L-4	TP, FN	AP	CuP with SBD	PA	NS	Bottleneck (shared amongst flows from similar connection) identification	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
MPTCP-based schemes								
SB-FPS [84] (2020)	L-4	TP	InD	CuP with SBD	NS	CnS	SB-FPS addresses the issue of bottleneck detection to maintain bottleneck fairness	General
CL-ADSP [29] (2019)	L-4, L-2 (CL)	TP, FTT, ETE-D, NRL	InD	CuP(OLIA)	PC	CnS	CL-ADSP addresses the issue of receiver buffer blocking in highly varying conditions of MANETs. CL-ADSP also addresses the issue of congestion-induced losses at middle-boxes (routers) and offers congestion detection at intermediate routers by actively monitoring average MAC layer retries	Wireless
Xue et al. [85] (2018)	L-4	TP	InD	CuP	NS	CnS	This scheme addresses the issue of receiver buffer blocking in a lossy heterogeneous network environment	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
Xu et al. [24] (2017)	L-4	TP, ADD, ReTx. No	InD	CuP	PC	CnS	This scheme addresses the problem of reordering and buffer blocking during multi-path transmission scenarios	General
Couple + [86] (2016)	L-4	TP	InD	NS	NS	CnS	Couple + addresses the unfair CC problem with MPTCP when MPTCP gets combined with network coding	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
FMTCP [56] (2014)	L-4	TP, GP, AD	InD	CuP	PC	CnS	Cui et al. [56] have comprehensively evaluated the degraded MPTCP throughput performance. Since any individual sub-flow experiences severe packet losses and delay distresses the performance of other parallel running sub-flows. Consequently, it leads to the issue of becoming an MPTCP connection bottleneck	General
OLIA [87] (2013)	L-4	TP	InD	CuP(OLIA)	PC	CnS	OLIA addresses the issue that LIA cannot provide receptiveness and RP feature simultaneously (i.e., LIA fails to offer Pareto-optimality)	General

Table 1 (continued)

Proposal (Year)[1]	Solution type (DML)[2]	Metric(s)[3]	Path types[4]	Congestion control type[5]	Flow control type[6]	Receiver buffer[7]	Objectives (problem addresses)[8]	Network environment
Standard MPTCP [20–22] (2014) (2013) (2011)	L-4	TP	InD	CuP(LIA)	PC(S-f)	CnS	The standard MPTCP architecture [20, 22] and later its CC scheme [21] address the issue of TCP Friendliness and have kept the RP feature in a multi-path communication system	General

[1] *CMT-SR* Selective Retransmission-based CMT, *CMT-CQA* Cross-layer QoS-Aware adaptive CMT, *W-PR-SCTP* Westwood SCTP with Partial Reliability, *W-SCTP* Westwood SCTP, *LS-SCTP* Load-Sharing SCTP, *BA-SCTP* Bandwidth Aggregation SCTP, *SB-FPS* Shared Bottleneck-Forward Prediction Scheduler, *OLIA* Opportunistic Linked-Increases Algorithm

[2] L-5: Application Layer, L-3: Network Layer, L-2: Data Link Layer, L-1: Physical Layer, *DML* Decision Making Layer, *CL* Cross-Layer

[3] *TP* ThroughPut, *ARD* Average Reordering Delay, *ATT* Average Transmission Time, *J* Jitter, *GP* GoodPut, *EtE-D* EtE-Delay, *PTP* Payload ThroughPut, *APDD*: Average Packet Delay Difference, *IPD* Inter Packet Delay, *V-PSNR* Video Peak Signal-to-Noise Ratio, *SSIM* Structural Similarity Index, *NRL* Normalized Routing Load, *ADD* Average Delivery Delay, *ReTx. No.* Retransmission Number, *FN* FairNess, *AD* Average Delay

[4] *AP* Alternate Path, *DP* Disjoint Path, *InD* InDependent, *SP* Similar Path

[5] *UC* UnCoupled, *CuP* CouPled, *CuP (OLIA)* CouPled with OLIA, *CuP (LIA)* CouPled with Linked-Increases Algorithm (LIA), *SBD* Shared Bottleneck detection

[6] *PA* Per Association, *PPB* Per Path Basis, *PA(R)* Per Association at Receiver, *PPB(S)*: Per Path Basis at Sender, *PC(S-f)* Per Connection (Sub-flow basis), *PC* Per Connection

[7] *UB* UnBounded, *NS* Not-Specified, *CnS* ConStrained

[8] *FR* Fast Retransmission, *DIO* Delay Insensitive Optimization, *CC* Congestion Control, *RP* Resource Pooling, *MANETs* Mobile Ad-hoc NETworks

4 The DB-CMT Scheme

The conventional CMT and CMT-PF exploit the RR data scheduling strategy to transmit data over multiple network paths. The RR data scheduling strategy initially sends an equivalent amount of data on each path. However, as the load on the network increases over time, the number of incoming ACKs via multiple paths will also get affected, causing multiple timeouts and retransmissions in the network. Due to this, the size of the cwnd will change continuously. Hence, different amounts of data can be sent by this scheduling scheme depending on the condition of each path. As a result, a slower path delivers fewer amounts of data than a faster path. Therefore, the receiver receives out-of-ordered data packets. As the frequency of unordered data packets increases, the receiver buffer gets blocked. Consequently, higher Negative ACKs (NACKs) get generated, further reducing the size of *cwnd* unnecessarily, reducing channel utilization, and ultimately leading to inferior throughput performance. Typically, each path's delay, bandwidth, and PLR can vary comprehensively in a multi-path environment. Therefore, for efficient multi-path communication, we need a data scheduling (or distribution) scheme to take care of the parameters mentioned above efficiently. Initially, we suggested an Adaptive data chunk scheduling policy for CMT (A-CMT) [71] which acclimatizes the transmission rate rendering to bandwidth and delay of the path. For this, we considered path delay as a foremost factor which certainly shows a path's current traffic status precisely because each path's delay constantly varies when the load on the path changes. Thus, we presented an A-CMT policy, and the purpose of this policy is to estimate the expected and the actual transmission rate accurately. In A-CMT, we assessed the variation between these delays and adapted the *cwnd* growth policy accordingly. This paper proposes a DB-CMT scheme consisting of a *cwnd* adaptation policy (originally from A-CMT). On top of A-CMT, we are suggesting two new approaches, which are as follows: (I). Path Selection Policy, and (II). Fast Retransmission Policy. The schemes (i.e., CMT-PF, CMT-QA, and A-CMT) implemented on top of CMT are having an issue with their fast retransmission policy. According to their fast retransmission policy, these schemes state that whenever there is a need to perform fast retransmissions, always perform fast retransmissions on a path with a larger *cwnd* size or ssthresh. This is because these policies assume that if a path's *cwnd* size is sufficiently larger than that of the other path, that path is running more seamlessly. And at that point of time (i.e., when a scheme requires fast retransmission), if the paths have the same *cwnd* sizes, all these policies randomly select a path and perform fast retransmission. Such random selection may inaccurately choose the path having the maximum PLR or minimum bandwidth and may cause more retransmissions, buffer blocking, and losses. So we believe that retransmission is also overhead in itself, and in this way if such random selections are made, it further severely affects the delivery performance of any scheduler. We propose modifying the conventional fast retransmission policy to handle this issue well.

To lessen the receiver buffer blocking issue, fast retransmission issue, and to improve the utilization of available bandwidth, DB-CMT incorporates the three new policies, which are as follows: (I). D-DSP, (II). RTX-CL, and (III). D-FRP. Figure 1 shows the design of DB-CMT, which contains an SCTP sender, SCTP receiver, and 'n' asymmetric wired communication paths. On the sender side, there are three vital DB-CMT modules (i.e., D-DSP, RTX-CL, and D-FRP), compound interfaces (I-1, I-2, …, I-N), and a sender buffer. The chunks will be reassembled and collected inside the receiver buffer at the receiver side if the router has done the fragmentation of data chunks. D-DSP aims to transmit data chunks over multiple network paths with respect to their conforming traffic load. D-DSP estimates a path load by utilizing their corresponding delay and *cwnd* parameters. Meanwhile, RTX-CL manages to lessen the absurdness of selecting the retransmission destination. Finally, D-FRP reduces the problem of the needless *cwnd* reductions due to unordered data chunk reception at the receiver side.

This work proposes a data chunk scheduling policy that transmits more data chunks through the minimum delay path that minimizes the out-of-order data chunk delivery. This scheduling policy estimates each path's actual and expected data transmission rate. The approximation of these mentioned rates utilizes $\mathrm{RTT_i}$, $cwnd_\mathrm{i}$, and minimum RTT ($\mathrm{RTT_{min\_i}}$) parameters of any ith path in the network.

$$\mathrm{Act_{Rate\_i}} = \frac{\mathrm{cwnd_i}}{\mathrm{RTT_i}} \tag{1}$$

$$\mathrm{Exp_{Rate\_i}} = \frac{\mathrm{cwnd_i}}{\mathrm{RTT_{min\_i}}} \tag{2}$$

$$\mathrm{D_i} = \left(\mathrm{Exp_{Rate\_i}} - \mathrm{Act_{Rate\_i}}\right) * \mathrm{RTT_{min\_i}} \tag{3}$$

where $\mathrm{Act_{Rate\_i}}$ is the actual transmission rate, $\mathrm{Exp_{Rate\_i}}$ is the expected transmission rate, and $\mathrm{D_i}$ stipulates the load on the ith path. When the traffic on the ith path surges, the value of $\mathrm{D_i}$ rises consistently. The proposed policy utilizes the $\mathrm{D_i}$ factor to schedule the data chunk over multiple paths.

4.1 D-DSP Module in DB-CMT

In our policy, the source approximates an expected and the actual transmission rate for each destination based on Eqs. (1) and (2). Then, based on the estimation of these rates, we have calculated the factor $\mathrm{D_i}$ using Eq. (3). Now, two threshold variables, λ, and δ, decide the path traffic intensity. Typically, whenever the path traffic intensity is high and the sender tries to increase the size of *cwnd*, it ultimately causes network congestion. Thus, the proposed approach uses λ and δ threshold variables to send data according to path traffic intensity. If the path traffic intensity is low, the λ threshold is triggered. Meanwhile, δ will be triggered when the path has a high traffic intensity. The initial values of λ and δ are chosen to render the threshold estimation scheme introduced in our preceding work

Table 2 Advantages and disadvantages of SCTP/CMT-based and MPTCP-based schemes

Proposal (Year)	Advantages	Disadvantages
SCTP/CMT-Based CMT Schemes		
CMT-VR [46] (2018)	CMT-VR advocates improved performance against existing CMT schemes by assuring transmission consistency with the help of raptor codes and a flexible retransmission policy. Hence, CMT-VR relies on utilizing a flexible chunk retransmission scheme to maintain transmission consistency, which increases the applicability and reliability of this scheme for the services based on real-time video transmission	There is a strict requirement to realize the inter-packet redundancy that CMT-VR fails to offer. This scheme requires sufficient power consumption and extra delay for incorporating an additive encoding/decoding procedure, which may cause CMT-VR as inefficient in terms of power consumption
CMT-SR [74] (2018)	CMT-SR suggests improved performance against existing CMT schemes by assuring transmission consistency with the help of a flexible retransmission policy. CMT-SR working mechanism is more appropriate and supple for real-time informal video application data delivery because it is less affected by the higher delay and more significant generation of duplicate chunks than other existing CMT approaches	CMT-SR scheme necessitates more modifications and alterations to the current implemented standard at the L-4 level. It requires significant alterations both at the sender and the receiver sides. Also, the scheme does not consider the case of dynamic paths' conditions. And here, it will be essential to know how this policy works in rapidly changing path conditions
Verma et al. [28] (2018), A-CMT [71] (2017)	These approaches suggest improved network performance by effectively identifying between a network congestion event or out-of-order delivery events at the destination and reducing the size of *cwnd* accordingly. Therefore, these approaches more appropriately utilize the channel than other CMT approaches	Today, global Internet traffic consists of more than 70% of TCP traffic. Hence, these existing CMT schemes might be unfair towards other parallel competing TCP flows
Arianpoo et al. [75] (2017), CMT-NC [38] (2016)	These network coding-based schemes alleviate the consequences of data reordering in the network. Both techniques effectively improve transmission management in terms of reliability and flexibility	These schemes focus only on the feedback information transferred using incoming ACK packets. In particular, both schemes' sender has no exact information concerning prior scheduling. As a result, both techniques cannot optimize the scheduling policy further

Table 2 (continued)

Proposal (Year)	Advantages	Disadvantages
CMT-CA [45] (2016)	CMT-CA incorporates required vital parameters (i.e., video content) in conventional SCTP and improves its chunk scheduler. This policy significantly assists CMT-CA in optimizing and maintaining the video quality	CMT-CA entirely relies on the chunk retransmissions notion to maintain transmission consistency, which ultimately reduces the applicability and reliability of this scheme for the services based on real-time video transmission
CMT-CQA [76] (2015)	CMT-CQA dynamically identifies the slowest network path, ultimately eliminating that path from the transmission. This idea significantly assists CMT-CQA in dodging the issue of aggregated bandwidth deprivation	CMT-CQA relies on the strict principle of utilizing 3-dup SACKs for loss recognition and salvage. Hence, a sender cannot retransmit a lost data chunk until its sequence number is informed as lost three times continuously. Consequently, this stern principle can affect the delivery performance of this scheme
Optimized MPSCTP [41] (2015)	Tx-CWND path selection policy and heuristic approach to implement DIO in MPSCTP have been suggested to improve the throughput and retransmission. Tx-CWND path selection policy minimizes the processing at the source node	Optimized MSTCP has not used the path quality as a factor to retransmit the data chunk
CMT-CL/FD [15] (2014)	This cross-layer-based proposal observes path quality precisely. In contrast to SCTP, it does not blindly half the *cwnd* size for any packet loss. It regulates the growth of *cwnd* more correctly. Specifically, it adapts *cwnd* growth by considering the cause behind such loss. As a result, it performs fine concerning fairness and video delivery	If the network experiences higher delays and loss rates, this scheme needs extra modifications. The operation of this scheme is based on the strict principle of ordered chunk reception restrictions. The scheme admits that the network is extremely suffered from the problem of reordering. As a result, the delivery performance can be affected
CMT-DA [44] (2014)	Suggested numerous notions regarding efficient *cwnd* adaptations, per-path prominence evaluation, modified retransmissions, and rate (flow) provisions assist CMT-DA in minimizing the video distortion	To maintain transmission consistency, CMT-DA entirely relies on chunk retransmissions which ultimately reduces the applicability and reliability of this scheme for the services based on real-time video transmission

Table 2 (continued)

Proposal (Year)	Advantages	Disadvantages
Min–Max MPSCTP [40] (2013), MPSCTP [39] (2011)	MPSCTP improves throughput performance by effectively reducing the average number of retransmissions compared with the traditional CMT scheme. Min–Max MPSCTP, an improvised version of MPSCTP, supports further improvement in performance by minimizing the packet ETE delay over numerous network paths	MPSCTP cannot perform well in a highly lossy environment where a packet or ACK gets lost due to channel contentions, channel errors, wireless interface properties, mobility, constrained battery power-induced route disconnections, and network congestion. Meanwhile, assuming there is no loss of ACK packets in the network highly limits the scope of the MPSCTP scheme in the wireless or highly lossy environment. At the same time, the Min–Max MPSCTP scheme suffers from resource utilization problems due to its equal data distribution policy
CMT-QA [37] (2012)	Suggested a path quality-based data distribution and retransmission policy that minimizes the receiver buffer blocking and improves the throughput	CMT-QA has not considered the time required by a path to deliver the data chunk
Adhari et al. [77] (2011)	Adhari et al. [77] suggested an improved buffer handling scheme that dynamically utilizes the pooled buffer storage. Subsequently, it improves the performance by enhancing the probability that a faster path can get more buffer storage and send more data to the network	Although, the scheme effectively increases the throughput performance of the system by maintaining proper stability of the number of outstanding bytes on all available network paths. However, the scheme treated the whole network as a black box and did not consider path quality and asymmetric characteristics
Dreibholz et al. [33, 34] (2010) and (2011)	This scheme can handle the varying asymmetric factors and parameters of available network paths by using the novel concepts of chunk/data re-scheduling, buffer splitting, and adaptive FR scheme	This scheme suffered from the local blocking problem during asymmetric path characteristics conditions. Additionally, the notion of a uniform size buffer for the available network paths eventually leads to the problem of wastage of resources in terms of buffer space on substandard paths

Table 2 (continued)

Proposal (Year)	Advantages	Disadvantages
CMT-PF [35, 78] (2009) (2008)	CMT-PF efficiently alleviates the buffer blocking issue of CMT by detecting the network paths with inferior quality and stating those paths as potential failed paths. Subsequently, CMT-PF can perform well in highly varying receiver buffer situations	CMT-PF does not contemplate the dynamic differences of path quality factors and ultimately utilizes the RR scheduling procedure. As a result, a slower path delivers fewer amounts of data than a faster path. Therefore, the receiver receives out-of-ordered data packets. As the frequency of unordered data packets increases, the receiver buffer gets blocked
W-PR-SCTP [79] (2007)	To circumvent longer delays, W-SCTP-PR suggests incapacitation of SCTP retransmissions. Hence, the W-SCTP-PR scheduler performs well in delay-sensitive real-time applications	W-SCTP-PR suggests the importance of throughput (transmission) performance enhancement. However, this scheme does not entirely deal with the scenario in which the available network paths encompassed in the SCTP association display highly varying RTTs
CMT [19] (2006)	Compared to other multi-path approaches, CMT assists by handling the reordering issue (i.e., redundant retransmission). Also, this scheme competently handles the excessive ACK traffic in a multi-path communication scenario by suggesting the concept of delayed ACK for out-of-order segments	In the case of a shared path with other paralleled running competing TCP flows, this scheme causes unfairness in the system. CMT does not contemplate the dynamic differences of path quality factors and ultimately utilizes the RR scheduling procedure. As a result, a slower path delivers fewer amounts of data than a faster path. Therefore, the receiver receives out-of-order data packets. As the frequency of unordered data packets increases, the receiver buffer gets blocked
W-SCTP [80] (2004)	W-SCTP schedules or balances the load over multiple available network paths by estimating the bandwidth availability of paths	The performance of W-SCTP gets severely affected in a different path characteristics environment. Also, this method does not openly deal with the issue of packet reordering, which is a common problem in the multi-path transmission

Table 2 (continued)

Proposal (Year)	Advantages	Disadvantages
LS-SCTP [81, 82] (2004)	LS-SCTP schedules or balances the load over multiple available network paths by estimating the bandwidth availability of paths. LS-SCTP contains a modified retransmission scheme that guarantees the fast transfer of data packets	LS-SCTP does not deal with the issue of buffer blocking at the receiver end
BA-SCTP [83] (2003)	BA-SCTP accomplishes a coupled CC strategy for the flows competing for a shared buffer. Consequently, BA-SCTP flows are highly fair with other competing TCP flows that are shared the same buffer	BA-SCTP requires extensive changes in the conventional standard to obtain the packet loss classification evidence at the receiver side, which is highly infeasible. Also, the method does not consider the impact of constrained receiver buffer on the throughput performance
MPTCP-based schemes		
SB-FPS [84] (2020)	SB-FPS suggests improved throughput performance than existing MPTCP CC approaches while sustaining appropriate fairness towards other TCP flows at a shared bottleneck	Cubic-TCP [88] is a widely implemented TCP variant because most Android and Linux operating systems use this variant as an L-4 protocol. Thus, the fairness with Cubic is another challenge for any new multipath TCP variant. An extensive SB-FPS evaluation is needed on the inter-protocol fairness parameter with Cubic-TCP. Also, SB-FPS may suffer from the problem of higher transmission delay and reduced transmission rate since it requires a disproportionate amount of time for traffic scheduling
CL-ADSP [29] (2019)	CL-ADSP utilizes delay variations and actively schedules the transmission on multiple available paths. Also, CL-ADSP takes the intermediate feedback (routers) and dynamically decisions on the following scheduling criteria. Hence, CL-ADSP more appropriately handles the asymmetric nature of the path than other existing MPTCP approaches	CL-ADSP utilizes RTT estimations for evaluating delay variations. When the RTT estimations are used as a network congestion metric, a lesser channel utilization may be attained in the existence of reverse traffic. Hence, CL-ADSP may not perform better during the presence of reverse traffic

Table 2 (continued)

Proposal (Year)	Advantages	Disadvantages
Xue et al. [85] (2018)	This scheme reduces the uncertainties caused during data scheduling by considering feedback via receiving SACKs. This suggested scheme eradicates the preceding scheduling deviation for the next scheduling round. Adopting this procedure assists this scheme in reducing the gathered parameters estimation inaccuracies and ultimately helps make accurate scheduling decisions	Wei et al. [84] observed that the scheme [85] treats each running sub-flow as a lone TCP, and the authors did not contemplate that all subflows' *cwnd* could be coupled in the CA phase to guarantee fairness. Hence, the scheme [85] cannot offer good performance in a shared bottleneck environment
Xu et al. [24] (2017) Couple + [86] (2016)	These policies alleviate the high coding delays and enhance the throughput performance by addressing the unfair CC and buffer blocking problem	These policies ignored the feedback information via SACKs. Consequently, the sender does not have any precise prior information concerning scheduling in these policies. Also, these policies do not have any other approach to further improving the scheduling scheme for the next level rounds
FMTCP [56] (2014)	In FMTCP, the changing properties of multiple paths are competently managed using fountain codes. The FMTCP scheme functions fine beneath fast changes in the quality of network paths	In the case of a real-time video application and rigid delay restrictions, the block size of the fountain code becomes too large, and thus it is unsuccessful
OLIA [87] (2013)	OLIA effectively approximates congestion status on individual sub-flow and ultimately diverges the load over the least congested network paths. OLIA commendably supports by making LIA Pareto-optimal in nature	The packet re-ordering delay factor causes degradation in the throughput performance of MPTCP, and OLIA does not contemplate this imperative factor. Meanwhile, OLIA does not consider the other reasons for packet loss despite network congestion. Since it leads to the issue of unnecessary adaptations in *cwnd* in wireless scenarios, OLIA's capability can be highly limited in rapidly changing wireless scenarios

Table 2 (continued)

Proposal (Year)	Advantages	Disadvantages
Standard MPTCP [20–22] (2014) (2013) (2011)	The LIA (Coupled CC) scheme implemented in Standard MPTCP ensures fairness in resource distribution on numerous available network paths and offers pooling (resource) among them	LIA suggested improved performance by controlling the aggressiveness of MPTCP sub-flows compared to contending TCP flows. Still, LIA fails to classify whether the sub-flows share identical or non-identical bottlenecks. Consequently, it leads to inferior throughput performance of the protocol

[71]. As shown in Eq. (4), if the D_i factor is smaller than that of λ, the path has smoother traffic. Consequently, the sender can increase the size of *cwnd* by one Maximum Transmittable Unit (MTU). Next, if the D_i factor is more significant than δ, the path has sufficient traffic. Hence, the sender keeps the size of *cwnd* as is. Conversely, suppose the Di factor is superior to that of λ and inferior to that of δ. In that case, this shows that the path is not carrying enough traffic, and there is a possibility that we can transmit some more traffic on that (shown in Algorithm 1). Consequently, the sender can increase the size of *cwnd* according to the growth factor shown in Eq. (4). This technique can control the aggressiveness of *cwnd* growth and subsequently offers adequate time to settle the network congestion condition. Also, this reduces the possibility of buffer blocking at the receiver end.

$$\mathrm{cwnd}_{\mathrm{i}+1} = \begin{cases} \mathrm{cwnd}_{\mathrm{i}} \quad \mathit{if} \;\; \mathrm{D}_{\mathrm{i}} > \delta \\ \mathrm{cwnd}_{\mathrm{i}} + \left(\frac{\mathrm{MTU}}{2}\right) \quad \mathit{if} \;\; \lambda < \mathrm{D}_{\mathrm{i}} < \delta \\ \mathrm{cwnd}_{\mathrm{i}} + \mathit{MTU} \quad \mathit{if} \;\; \mathrm{D}_{\mathrm{i}} < \lambda \end{cases} \tag{4}$$

Algorithm 1: D-DSP Module Algorithm

```
For each SACK received (at sender side for an individual path):
Requirement: RTT, $RTT_{min}$, MTU, cwnd, λ, δ
Initialization: MTU=1500 Bytes, λ=1, δ=3
Begin
Estimate $Act_{Rate\_i}$ by using Eq. (1)
Estimate $Exp_{Rate\_i}$ by using Eq. (2)
Estimate $D_i$ by using Eq. (3)
If ($D_i < λ$) then
    $cwnd_{i+1} = cwnd_i + MTU$
Else If ($D_i > δ$) then
    $cwnd_{i+1} = cwnd_i$
Else
    $cwnd_{i+1} = cwnd_i + MTU/2$
End If
End
```

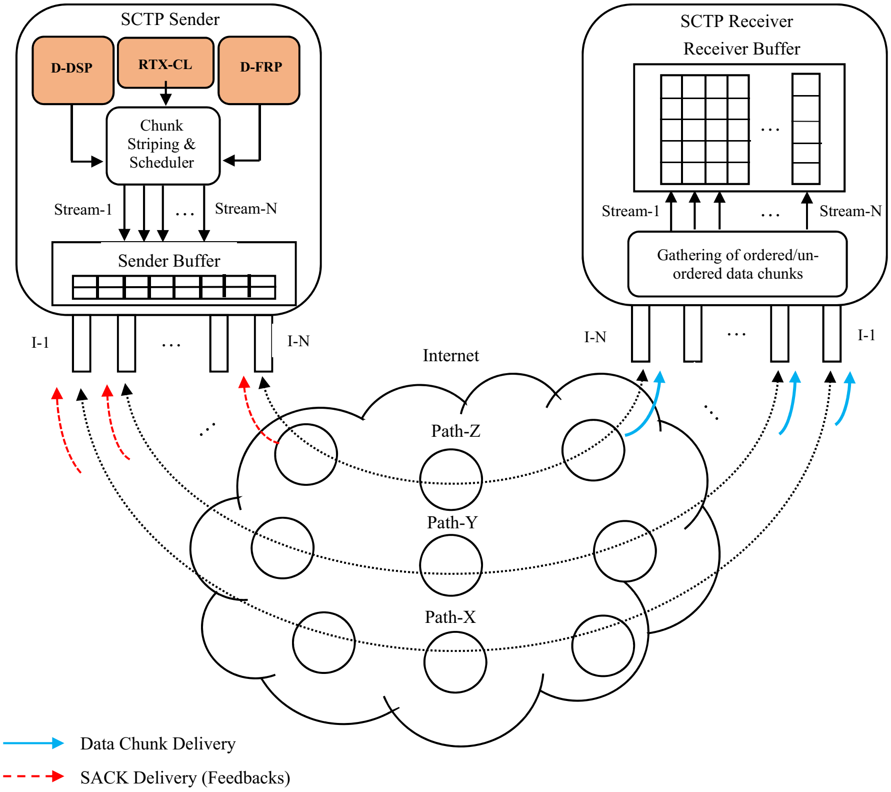


Fig. 1 The DB-CMT design

4.2 RTX-CL Module for Path Selection Policy in DB-CMT

Iyengar et al. [19] initially proposed five path selection policies to decide the data chunk retransmission path. RTX-SSTHRESH and RTX-CWND select paths with either maximum *cwnd* or maximum ssthresh factors. If numerous paths have identical *cwnd* or ssthresh, these schemes can randomly choose the retransmission path. In such a random selection process, there is a possibility to select the path with low QoS parameters, such as low available bandwidth and higher loss rate. To solve this problem, we propose the RTX-CL path selection policy that chooses the best retransmission path according to the highest *cwnd* and lower loss rate. Initially, RTX-CL selects the retransmission path with the highest *cwnd*. If multiple paths have the same highest *cwnd*, it checks the lowest loss rate amongst the destinations chosen in the previous steps. If the policy cannot find one path, then RTX-CL selects one path randomly. The algorithm for the proposed RTX-CL is shown in Algorithm 2. To verify the RTX-CL policy, we designed, implemented, and analyzed the scheme on *ns*-2.35 [72]. For this, we compare the performance of the RTX-CL retransmission path selection scheme against the RTX-CWND and RTX-LOSSRATE policies.

The network topology utilized for the simulation is shown in Fig. 2. In this considered topology, PATH-1 has a fixed PLR of 1%, while the PLR of another path (i.e., PATH-2) varies between 1 and 10%. Meanwhile, we have shown the considered values for delay and bandwidth of both paths in Fig. 2. The SCTP sender S has S1 and S2 interfaces, while SCTP receiver D has D1 and D2 interfaces. The SCTP sender S transmits the FTP traffic to receiver D. The receiver buffer size is 64 KB (default), and the link queue has been configured with a drop-tail queuing policy of 50 packets size.

Algorithm 2: Fast retransmission and RTX-CL path selection policy

```
For every SACK received (at sender side for each path):
Requirement: RTT, $RTT_{min}$, MTU, cwnd, ssthresh
Initialization: MTU = 1500 Bytes
Begin
$\omega_i = (RTT_i - RTT_{min\_i})/RTT_i$
If (four duplicate SACK received)
    $ssthresh_{i+1} = MAX(cwnd_i - (cwnd_i \times \omega_i), 4 \times MTU)$
    $cwnd_{i+1} = ssthresh_i$
End If
// RTX-CL path selection policy
L_cwnd = 0 // For largest cwnd
L_lossrate = 1000 // For lowest loss rate
For i=0 to n
    If (L_cwnd < $Path_i$)
        L_cwnd = $cwnd_i$                // For i^th path cwnd
        Set $Path_i$ as a retransmission destination
    Else If (L_cwnd == $cwnd_i$)         // Multiple paths have the same cwnd
        For j=0 to n
            If (L_lossrate > $Path_j$)
                L_lossrate = $lossrate_j$    // For j^th path loss rate
                Set $Path_j$ as a retransmission destination
            Else If (L_lossrate==$lossrate_j$)    // Multiple paths have the same loss rate
                Select the path randomly
            End If
        End For
    End If
End For
    Retransmit the data chunk on the selected path
End
```

Figure 3 shows the throughput performance of different retransmission path selection policies with variable PLRs. Here, DB-RTX-CL, DB-RTX-CWND, and DB-RTX-LOSSRATE use the proposed DB-CMT, while CMT-RTX-CWND and

CMT-RTX-LOSSRATE use the customary CMT scheme. As discussed earlier, RTX-CL selects the initial retransmission path with a larger *cwnd* size. If every path has the same *cwnd*, this method chooses a path with the minimum PLR. However, RTX-CWND and RTX-LOSSRATE select the retransmission path based on the largest *cwnd* and smallest PLRs.

It is evident from the results (shown in Fig. 3) that as the PLR increases, the throughput performance of all retransmission path selection (RTX) policies declines significantly. In this analysis, CMT-RTX-CWND outperforms CMT-RTX-LOSSRATE, while the DB-RTX-CL achieves higher throughput performance than other retransmission policies. From Fig. 4, it is evident that DB-RTX-CL has fewer average retransmission timeouts than other retransmission policies. Figures 3 and 4 confirm that RTX-CL can be an optimal retransmission policy for networks with higher PLRs.

4.3 D-FRP Module in DB-CMT

In the fast retransmission phase, every time CMT obtains four duplicate SACKs, it assumes this event as congestion. Subsequently, CMT reduces the aggressiveness of *cwnd* growth by blindly halving it out of the current *cwnd* size. But this approach of *cwnd* reduction is inappropriate because duplicate SACKs are also received when the destination gets unordered data chunks- the frequent reductions in the *cwnd* lead to a problem of compromised throughput performance. When congestion surges, the RTT factor rises, whereas the unordered data chunk delivery may not increase the RTT. If we comprise delay (path) as a significant factor in cwnd reduction, it will appropriately regulate the reduction in *cwnd* growth instead of carelessly reducing it to half. Therefore, the proposed method comprises the up-to-date path's *cwnd* and RTT as a *cwnd* lessening factor. This factor is self-sufficiently estimated for each path while getting four duplicate SACKs. Eq. (5) shows the proposed *cwnd* reduction policy.

$$\mathrm{cwnd}_{\mathrm{i+1}} = \mathrm{cwnd}_{\mathrm{i}} - \frac{\mathrm{cwnd}_{\mathrm{i}} * (\mathrm{RTT}_{\mathrm{i}} - \mathrm{RTT}_{\mathrm{min\_i}})}{\mathrm{RTT}_{\mathrm{i}}} \tag{5}$$

The $(RTT_i—RTT_{min\_i})/RTT_i$ provides a constant factor (ω) that changes in accordance with RTT_i variation. The value of ω will be significant when RTT_i variation is large, and if RTT_i variation is smaller, ω will be small. Now, in Eq. (5), we estimate the next *cwnd* by using $cwnd*\omega$. The $cwnd*\omega$ shows the extra buffer consumed by the current flow on the ith path. Therefore, when congestion is detected due to packet drop or buffer blocking, the source reduces the *cwnd* according to the extra buffer consumed by the current flow on the ith path. As a result, Eq. (5) reduces the size of *cwnd* to a lesser amount when ω is small. As ω increases, the reduction in *cwnd* also increases. The ω lessens the size of *cwnd* of the current path by a significant amount if congestion occurs, while *cwnd* decreases with the trivial amount during the situation of unordered data chunk delivery (shown in Algorithm 2). The

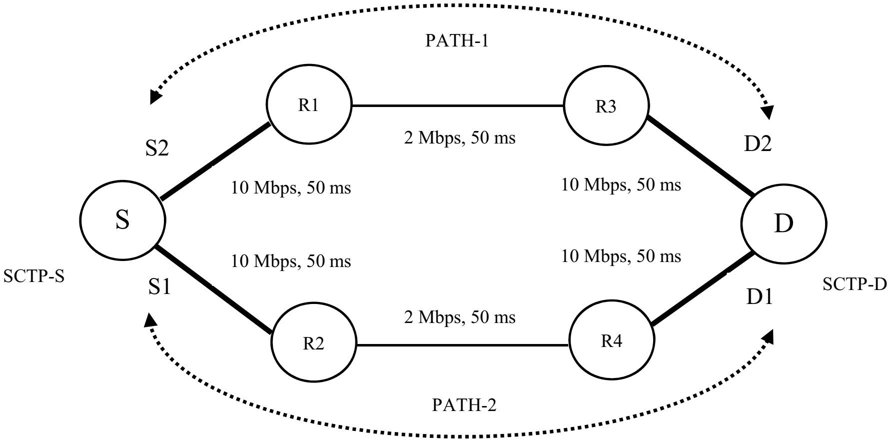


Fig. 2 Network topology

selection of the retransmission path is also made according to the proposed RTX-CL path selection policy.

5 Performance Evaluation

We performed simulation experiments with some imperative CMT variants. Section 5.1 presents the network and topological arrangements for the simulated CMT variants. In Section 5.2, we present and discuss the performance evaluation of DB-CMT with CMT [19], CMT-PF [35], CMT-QA [37], and A-CMT [71] using *ns*-2 on different sets of bandwidth values scenarios. Further, *ns*-2 includes the latest CMT module developed initially and modeled by the University of Delaware, US.

5.1 Simulation Environment

We have realized two types of network environments to evaluate the performances of all the CMT schemes, which are as follows: (I). Network environment where bandwidth is sufficiently available (i.e., CASE-1) and (II). Network environment where bandwidth is extremely limited (i.e., CASE-2). Our main objective behind realizing such environments is that we will be able to assess the performance of all the schemes in both kinds of situations in a much better way. Figure 5 displays the network topology used for our performance assessment. This network structure has an SCTP sender having S1 and S2 interfaces and an SCTP receiver with D1 and D2 interfaces. Along with this, we have configured two routers (denoted R-1 and R-3) which are single-homed devices. To simulate severe congested circumstances at these routers, we also configure cross-traffics. We inject multiple cross-traffics to simulate the background traffic for heavy data exchange. This cross-traffic is produced by a Constant Bit Rate (CBR) generator attached over a UDP connection. These end-hosts interconnect through two disjoint paths (PATH-1 and PATH-2). The parameter configurations of both paths are listed in Table 3. Initially, the parameter settings of both paths are default and similar; hence, we change some specific parameters of PATH-2 at a time while keeping the parameters associated with PATH-1 constant. Consequently, we configure dissimilar paths' characteristics, and further, we evaluate the individual impact of parameters on the performance of the simulated schemes. All the simulation results presented are estimated by normalizing the results over several runs, which makes the consequence of the loss rate and cross-traffic on different simulated schemes more accurate and not affected by any other stochastic factors.

To simulate CASE-1, we have configured 10% fixed PLR on PATH-1 and variable PLR (which varies between 0 and 10%) on PATH-2. In particular, the delay and bandwidth of each attached link are shown in Fig. 5 as BandWidth (BW_{IJ}), Delay (D_{IJ}) (where IJ are the corresponding nodes to which a link is directly connected), and their actual values are presented in Table 4 for this simulation setup. However, to simulate CASE-2, we have configured a 1% fixed PLR on PATH-1 and the variable PLRs (varies between 1–10%) on PATH-2. The corresponding delay and bandwidth values for this simulation setup are shown in Table 4. Additionally, the source (SCTP) is attached with an FTP traffic generator, and the simulation time of this setup is 200 secs. Also, this topology has two UDP senders, U1 and U2, and two UDP receivers, U11 and U22. The U1 and the U11 are attached to R1 and R4, while U2 and U22 are attached to R2 and R3 routers. The UDP sources U1 and U2 are connected with the CBR traffic agent, while the SCTP source is attached to an FTP traffic agent. This simulation setup is organized with a drop-tail queuing scheme, and the default queue magnitude is 50 packets. DB-CMT uses the proposed RTX-CL in this simulation arrangement, while CMT and CMT-PF are organized with an RTX-CWND retransmission scheme. We do not comprise enormously low traffic throughout the simulations. Table 3 shows the other necessary simulation parameters utilized for the performance assessment.

5.2 Simulation Results and Discussions

This section presents the performance evaluation of DB-CMT against CMT, CMT-PF, A-CMT, and CMT-QA based on metrics such as average throughput (kbps), *cwnd* size variations, and FTT. In Sect. 5.2.1, we examine the performance and behaviour of DB-CMT where bandwidth is sufficiently available (CASE-1). In Sect. 5.2.2, we investigate the performance of DB-CMT in a network environment where bandwidth is highly limited (CASE-2), and finally in Sect. 5.2.3, we perform the statistical analysis for the simulated CMT schemes.

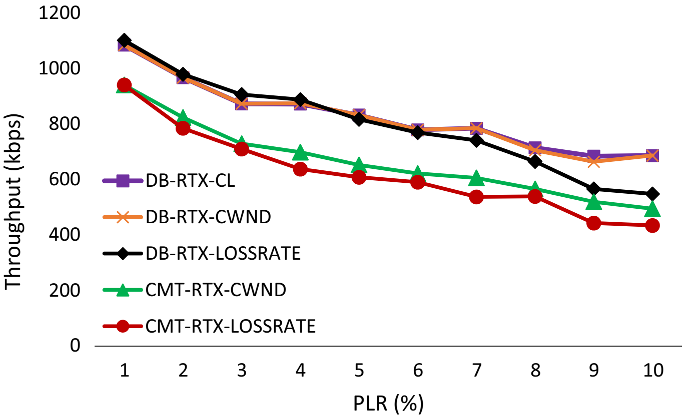


Fig. 3 Throughput performance of CMT and DB-CMT using different RTX policies

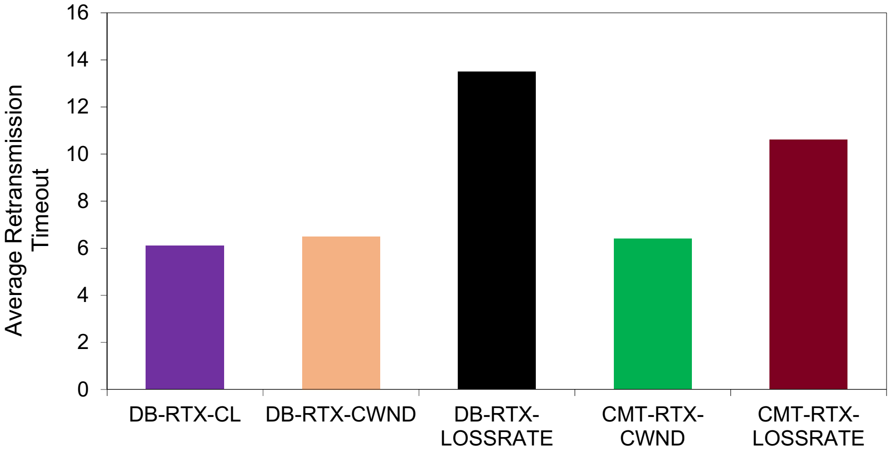


Fig. 4 Average retransmission timeout of CMT and DB-CMT with different RTX policies

5.2.1 Experimental Analysis of All Simulated CMT Schemes on a Sufficiently Available Bandwidth Network Environment

We investigate the performance of DB-CMT when bandwidth is sufficiently available in a network in terms of throughput with a 1 Mbps UDP background traffic rate. Figure 6 shows the variations in average throughput (kbps) for the entire simulation period. This experiment aims to confirm the competence of all the simulated CMT schemes to efficiently handle packet losses, which substantially impact the throughput performance. As shown in Fig. 6, as the PLR increases, throughput continually decreases for all the simulated schemes. The drop in throughput performance of CMT, CMT-PF, CMT-QA, A-CMT, and DB-CMT is around 74.31%, 75.80%, 73.67%, 72.48%, and 71.68% respectively. At a PLR of 1%, DB-CMT yields 11.33%, 13.30%, 7%, and 5.31% improved throughput performance against CMT, CMT-PF, CMT-QA, and A-CMT, respectively. While at a maximum PLR of 10%, DB-CMT's throughput performance is 27.09%, 29.83%, 15.10%, and 9.03% higher than CMT, CMT-PF, CMT-QA, and A-CMT, respectively. While at an average PLR of 5%, DB-CMT's throughput performance is 6.23%, 18.11%, 10.77%, and 7.07% higher than that of CMT, CMT-PF, CMT-QA, and A-CMT, respectively.

In particular, as the PLR increases, the chances of higher *cwnd* growth reductions increase. It increases the probability of higher transmission delay as well. Here, the throughput performance of CMT drops seriously on increasing PLRs (i.e., 1773 kbps to 455 kbps) because it reduces its *cwnd* size immediately as soon as it senses the packet loss. However, CMT-PF also does not offer substantial performance (i.e., 1843 kbps to 446 kbps) because it cannot precisely recognize packet drop due to short-term route failures in most cases. Also, CMT and CMT-PF utilize an RR-based scheduling scheme without considering the path quality factors. Hence, these schemes cannot efficiently utilize bandwidth and offer around 888 kbps and 867 kbps average throughput. In fact, initially, the scheduling scheme incorporated in CMT and CMT-PF schedules and transmits the same amount of data on all available network paths without considering any quality factors. However, CMT-QA offers 6.19% and 8.76% improved performance than CMT and CMT-PF because it can detect packet losses and congestion-induced path failure well on time and further assist the scheme in dynamically adapting its scheduling decisions. Moreover, A-CMT suggests 8.89%, 11.53%, and 2.54% improved performance against conventional CMT, CMT-PF, and CMT-QA schemes. This is because A-CMT utilizes the paths' quality factors (i.e., delay and bandwidth) as a crucial aspect of chunk scheduling and schedules more chunks on a path having lower delay and higher available bandwidth. Furthermore, DB-CMT outclasses CMT and CMT-PF schemes by offering 16.21% and 19.03% improved performances and suggests 9.43% and 6.72% better performance than CMT-QA and A-CMT. DB-CMT offers such performance because it eradicates the randomness in selecting a path during the fast retransmission procedure when the paths possess the same *cwnd* sizes at any point of time. Such random selection inaccurately chooses the path in CMT, CMT-PF, CMT-QA, and A-CMT schemes having the maximum PLR and minimum bandwidth and causes more retransmissions, buffer blocking, and losses. Meanwhile, DB-CMT also uses a delay and bandwidth-aware data chunk scheduling policy to distribute data over multiple paths, further assisting the scheme in effectively scheduling data over qualitative paths.

Afterward, we apply the concept of the confidence interval to estimate the average throughput and FTT (evaluated parameters 'e_p') performance variations via Eq. (6).

$$\mathrm{P_r}(\overline{\mathrm{M}} - \mathrm{Z}_{(1-\delta/2)} * (\mathrm{SD}/\sqrt{n}) \le e_p \le \overline{\mathrm{M}} + \mathrm{Z}_{(1-\delta/2)} * (\mathrm{SD}/\sqrt{n})) = 1 - \delta \qquad (6)$$

where $\overline{\mathrm{M}}$ is the sample mean, SD is the sample standard deviation, n is the sample size, '1-δ' is the confidence level, and '$\mathrm{Z}_{(1\text{-}\delta/2)}$' is a function of '1-δ/2', which can be refer through Table 5.

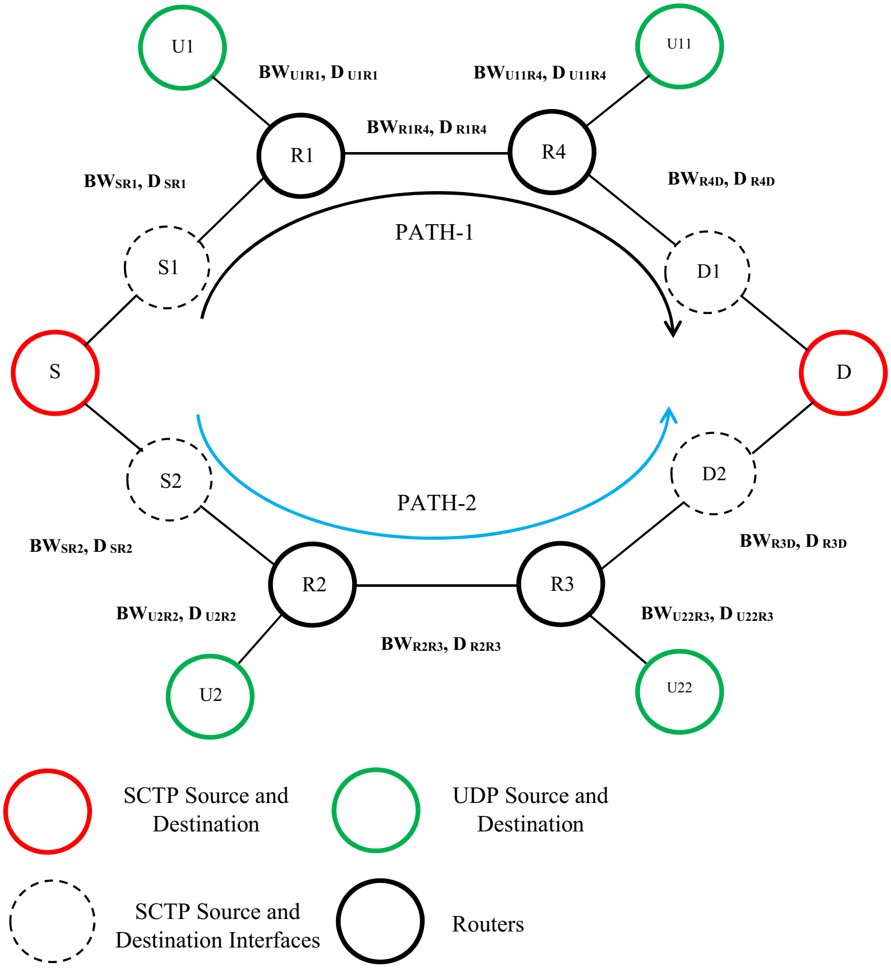


Fig. 5 The network topology

Besides, we also evaluate the confidence intervals on the considered parameters for the simulated schemes. Though to reduce their difficulty level in observation, the values of those intervals are not only shown in Fig. 7, but we have also added the table (Table 6) explicitly. In Table 6, we show the confidence intervals with a 95% confidence level concerning simulation results shown in Fig. 6. Also, each simulation point underwent repetition twenty times, and we confirmed the reliability of the obtained simulation results with a 95% confidence level. Figure 7 depicts the plots showing the average throughput performance and the conforming 95% confidence intervals for changing PLR.

Figure 8A–E shows the variations in *cwnd* growth for the entire duration of the simulation. The results have been obtained for CMT, CMT-PF, CMT-QA, A-CMT, and DB-CMT, and these results specify the amount of traffic (in terms of bytes)

Table 3 Simulation parameters

Parameter	Values
SCTP maximum segment size (MTU)	1500 bytes
SCTP data chunk size	1468 bytes
SCTP receiver buffer size (default)	64 KB
SCTP sender buffer size (default)	64 KB
SCTP application	FTP
SCTP RTX policy	RTX-CWND
Queuing policy	Drop-tail
Queue size	50 packets
Path PLR	PATH-1: 10%, PATH-2: 0–10% (CASE-1) PATH-1: 1%, PATH-2: 1–10% (CASE-2)
Simulation time	200 secs
Background traffic	UDP
UDP application	CBR (PATH-1 and PATH-2: 1 Mbps) (CASE-1) CBR (PATH-1:150 kbps, PATH-2: 400 kbps) (CASE-2)

over network paths (PATH-1 and PATH-2). To perform this analysis, we set 2% PLR on PATH-1 and 10% on PATH-2. In general, when we consider all the cases (Fig. 8A–E), we can comprehend that all the simulated CMT schemes have an initial *cwnd* size of 4380 bytes. This is because the traditional CMT scheme suggests a policy (i.e., $\text{cwnd}_{\text{Initial}} = \min$ (4*MTU, max (2*MTU, 4380 bytes))) to initialize *cwnd* initially. Also, the implementation of CMT schemes (i.e., CMT-PF, CMT-QA, A-CMT, and DB-CMT) given herein follow the traditional CMT implementation. It is evident from results (Fig. 8A–E) that the size of *cwnd* growth on PATH-1 is much larger than on the PATH-2 because the PLR on PATH-1 is smaller than the PATH-2. Figures 8A and B show the variations in *cwnd* growth for CMT and CMT-PF schemes. In CMT, the average *cwnd* growth on PATH-1 and PATH-2 is about 12845 bytes and 6306 bytes. In CMT-PF, the average *cwnd* growth on PATH-1 and PATH-2 is around 12361 bytes and 6042 bytes. In CMT-QA, the average size of *cwnd* growth on PATH-1 and PATH-2 is nearby 15585 bytes and 8685 bytes. In A-CMT, the average *cwnd* growth on PATH-1 and PATH-2 is near 14197 bytes and 7775 bytes. While, in DB-CMT, the average size of *cwnd* growth on PATH-1 and PATH-2 is nearby 16965 bytes and 9606 bytes. These results signify that CMT and CMT-PF schemes fail to achieve appropriate channel utilization when PLR on a path gets severe. DB-CMT, CMT-QA, and A-CMT suggest improved channel utilization performance on CMT and CMT-PF. Specifically, if we consider the case of PATH-2, which has 10% PLR, CMT-QA offers 37.70% and 43.74% improved performance than CMT and CMT-PF (shown in Fig. 8C). In the same scenario, A-CMT suggests 23.27% and 28.68% improved performance than CMT and CMT-PF (shown in Fig. 8D). However, DB-CMT offers 16.34% and 4.50% improved performance than CMT-QA and A-CMT. DB-CMT outperforms CMT and CMT-PF by providing

Table 4 Bandwidth and delay configurations

Notation	BW_{IJ}, D_{IJ}
Configured parameters for simulating network environment where bandwidth is sufficiently available (CASE-1)	
BW_{SR1}, D_{SR1}/BW_{R4D}, D_{R4D}/BW_{SR2}, D_{SR2}/BW_{R3D}, D_{R3D}	100 Mbps, 1 ms
BW_{R1R4}, D_{R1R4}/BW_{R2R3}, D_{R2R3} (Bottleneck Links)	5 Mbps, 45 ms
BW_{U1R1}, D_{U1R1}/BW_{U11R4}, D_{U11R4}/BW_{U2R2}, D_{U2R2}/BW_{U22R3}, D_{U22R3}	10 Mbps, 45 ms
Configured parameters for simulating network environment where bandwidth is extremely limited (CASE-2)	
BW_{SR1}, D_{SR1}/BW_{R4D}, D_{R4D}/BW_{SR2}, D_{SR2}/BW_{R3D}, D_{R3D}	0.75 Mbps, 50 ms
BW_{R1R4}, D_{R1R4}/BW_{R2R3}, D_{R2R3} (Bottleneck Links)	0.5 Mbps, 50 ms/0.75 Mbps, 30–50 ms
BW_{U1R1}, D_{U1R1}/BW_{U11R4}, D_{U11R4}/BW_{U2R2}, D_{U2R2}/BW_{U22R3}, D_{U22R3}	1 Mbps, 20 ms

52.30% and 59.01% improved performance. Also, DB-CMT suggests 10.65% and 23.54% better channel utilization than CMT-QA and A-CMT (shown in Fig. 8E).

Figure 9 shows the change in throughput (kbps) for the entire duration of the simulation. The CMT, CMT-PF, CMT-QA, A-CMT, and DB-CMT policy results indicate the traffic over simulated paths (i.e., PATH-1 and PATH-2) when a 64 KB receiver buffer is used. In addition, the PLR of PATH-1 and PATH-2 is 10% and 0%, respectively. Initially, the throughput of all the CMT schemes increases rapidly because all the policies quickly probe the available channel capacity. And, the Slow-Start algorithm in all the schemes doubles the *cwnd* size repetitively. We can observe variations in throughput for all the simulated CMT schemes due to congestion and buffer blocking induced packet losses. Then it recovers after *cwnd* amendments and fast retransmissions. When we compare DB-CMT with the rest of the simulated CMT schemes, it is clear that DB-CMT tolerates packet loss better and utilizes the obtainable aggregate bandwidth competently. For a particular instance, after 200 s of simulation time with a 64 KB receiver buffer, DB-CMT offers 13.87%, 9.40%, 6.45%, and 4.78% improved average throughput than CMT, CMT-PF, CMT-QA, and A-CMT, respectively.

In the next simulation, we investigate the effects of asymmetric loss rate on FTT performance. To perform this analysis, we have set 2% PLR on one path (PATH-1) and 10% on the other (PATH-2). In contrast, other important simulation configurations remain identical, as shown in Tables 3, 4, and Fig. 5. Figure 10 shows the results plots showing the average FTT (s) performance and the conforming 95% confidence intervals (shown in Table 7) for changing file sizes. These results show that as the file size increases, the FTT also upsurges. These results clearly show that the FTT is highest in the case of the CMT compared to other schemes. In the case of CMT, the average FTT (concerning changing file sizes) is around 229.17 s, while it is around 217.76 s, 216 s, 212.45 s, and 209.47 s in CMT-PF, CMT-QA, A-CMT, and DB-CMT, respectively. Due to its conventional scheduling scheme, CMT takes more time to transmit each size file. However, in this scenario, CMT-PF suggests some improvement in average FTT

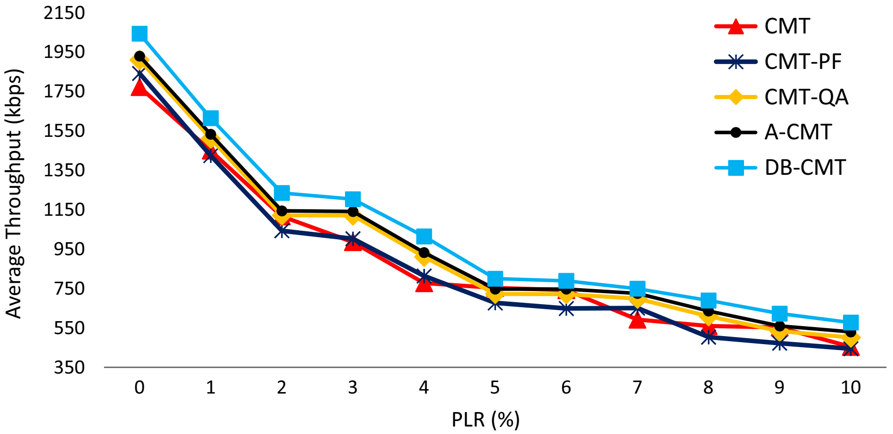


Fig. 6 The average throughput performance comparison between simulated CMT schemes with respect to varying PLRs

(i.e., around 5.23% lesser than the CMT) because it recognizes packet drop as a result of short-term route failures to some extent. Moreover, CMT-QA suggests comparable performance with respect to CMT-PF and A-CMT. Further, DB-CMT takes the minimum time to transmit each file size compared to other simulated schemes. DB-CMT takes 9.80%, 4.07%, 3.15%, and 1.68% lesser average FTT then CMT, CMT-PF, CMT-QA, and A-CMT respectively. DB-CMT takes 26.19% and 19.04% less time to transmit a 10 MB file than CMT and CMT-PF. While for the same scenario, DB-CMT suggests 9.52% and 7.14% lesser average FTT performance than CMT-QA and A-CMT. Moreover, DB-CMT takes 9% and 4.26% less time to transmit an average file size of 50 MB than CMT and CMT-PF. DB-CMT suggests 3.79% and 1.89% lesser FTT for the same scenario than CMT-QA and A-CMT. However, for the case of a maximum file size of 90 MB, DB-CMT suggests 8.20%, 2.91%, and 2.90% lesser FTT performance against CMT, CMT-PF, and CMT-QA, respectively, and suggests comparable performance (i.e., 1.08% lesser FTT) against A-CMT scheme. For minimum file sizes (i.e., ranges from 10 to 40 MB), the average FTT performances of CMT, CMT-PF, CMT-QA, A-CMT, and DB-CMT are around 116.87 s, 111.42 s, 109.23 s, 107.10 s, and 105.15 s, respectively. For maximum file sizes (i.e., ranges from 60 to 90 MB), the average FTT performances of CMT, CMT-PF, CMT-QA, A-CMT, and DB-CMT are around 341.25 s, 323.66 s, 322.30 s, 317.25 s, and 313.30 s, respectively. Also, we estimate the confidence interval for these results.

5.2.2 Experimental Analysis of the Simulated CMT Schemes on a Limited Bandwidth Network Environment

Figures 11, 12, 13, 14 compare the throughput performance of DB-CMT with CMT, CMT-PF, and CMT-QA for receiver buffer sizes 32 KB, 64 KB, and 128 KB, respectively. This simulation study demonstrates that the receiver buffer size affects

Table 5 Confidence level and equivalent Z score

Confidence level (%)	Z score ($Z_{(1-\delta/2)}$)
50	0.674
80	1.282
90	1.645
95	1.960
98	2.326
99	2.576

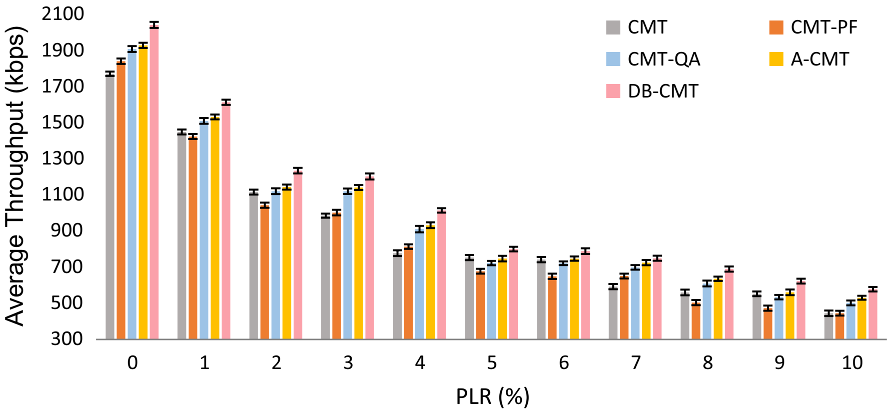


Fig. 7 Average throughput of all the simulated CMT variants with 95% confidence interval

the performance of CMT variants. It shows that as a receiver buffer size increases, the throughput of CMT variants increases. Figure 11 shows the throughput variations with the PLR when the receiver buffer size is 32 KB. CMT and CMT-PF show a linear throughput deviation because both methods use the RR-based data chunk scheduling policy. However, CMT-QA uses the path quality-based data distribution policy. Consequently, it achieves a better result compared to CMT and CMT-PF. Meanwhile, DB-CMT uses a delay and bandwidth-aware data chunk scheduling policy to distribute data over multiple paths. It is evident from Fig. 11 that DB-CMT performs better than the CMT, CMT-PF, and CMT-QA at all PLR values. At a minimum receiver buffer size (32 KB) case, DB-CMT's overall throughput performance is 7.45%, 5.93%, and 2.45% higher than CMT, CMT-PF, and CMT-QA, respectively.

In the 64 KB receiver buffer size case, DB-CMT's average throughput is 13.42%, 12.04%, and 8.07% more than CMT, CMT-PF, and CMT-QA, respectively. While at a maximum receiver buffer size (128 KB) case, DB-CMT's overall throughput performance is 15.05%, 15.26%, and 2.38% higher than that of CMT, CMT-PF, and CMT-QA, respectively (shown in Figs. 12 and 13). In Table 8, we show the confidence intervals with a 95% confidence level with respect to simulation results in Figs. 11, 12, 13. Also, each simulation point underwent repetition twenty times, and we confirmed the results' reliability with a 95% confidence level. Figure 14 depicts

Table 6 Analysis of the evaluated schemes via statistical parameters based on average throughput (kbps)

Metrics/statistical parameters	Evaluated schemes										
PLRs	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
CMT											
Average Throughput (kbps) after 20 iterations	1773	1451	1117	987	779	754	743	593	561	553	445
Sample Standard Deviation	26.03	29.42	28.89	24.62	34.05	30.59	32.15	31	33.84	27.8	32.5
Margin of error	12.18	13.77	13.52	11.52	15.93	14.32	15.05	14.51	15.84	13.01	15.21
Confidence interval width	24.36	27.54	27.04	23.04	31.86	28.64	30.1	29.02	31.68	26.02	30.42
Upper confidence interval (Max)	1785	1465	1131	998	795	769	758	608	577	566	461
Lower confidence interval (Min)	1761	1437	1104	975	763	740	728	579	545	540	430
CMT-PF											
Average throughput (kbps) after 20 iterations	1843	1425	1044	1004	815	678	650	652	505	474	446
Sample standard deviation	31.76	29.63	30.81	32.5	26.83	28.86	31.76	28.53	30.58	30.55	28.69
Margin of error	14.86	13.87	14.42	15.21	12.56	13.51	14.87	13.35	14.31	14.3	13.43
Confidence interval width	29.72	27.74	28.84	30.42	25.12	27.02	29.74	26.7	28.62	28.6	26.86
Upper confidence interval (Max)	1858	1439	1059	1019	828	692	665	665	519	488	459
Lower confidence interval (Min)	1828	1412	1030	988	802	665	635	639	491	460	432
CMT-QA											
Average throughput (kbps) after 20 iterations	1911	1512	1122	1122	912	724	723	700	611	535	503
Sample standard deviation	34.44	34.54	34.01	32.63	38.1	25.72	22.11	27.36	34.36	26.36	28.25
Margin of error	16.12	16.17	15.92	15.27	17.83	12.04	10.35	12.81	16.08	12.34	13.22
Confidence interval width	32.24	32.34	31.84	30.54	35.66	24.08	20.7	25.62	32.16	24.68	26.44
Upper confidence interval (Max)	1927	1528	1138	1137	930	736	733	712	627	547	516
Lower confidence interval (Min)	1894	1496	1106	1107	894	712	712	687	594	523	490
A-CMT											
Average throughput (kbps) after 20 iterations	1930	1534	1145	1142	933	749	748	726	637	561	531
Sample standard deviation	30.73	28.8	30.03	29.18	34.47	32.8	26.31	30.83	25.65	33.45	23.51

Table 6 (continued)

Metrics/statistical parameters	Evaluated schemes										
PLRs	0%	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
Margin of error	14.38	13.48	14.05	13.66	16.13	15.35	12.31	14.43	12.01	15.65	11
Confidence interval width	28.76	26.96	28.1	27.32	32.26	30.7	24.62	28.86	24.02	31.3	22
Upper confidence interval (Max)	1945	1547	1159	1156	950	764	761	741	649	577	542
Lower confidence interval (Min)	1916	1520	1131	1129	917	733	736	712	625	546	520
DB-CMT											
Average throughput (kbps) after 20 iterations	2044	1615	1236	1204	1016	801	790	750	691	624	579
Sample standard deviation	35.93	30.34	33.23	36.04	26.48	27.65	33.85	30.2	30.77	28.67	26
Margin of error	16.82	14.2	15.55	16.87	12.39	12.94	15.84	14.14	14.4	13.42	12.17
Confidence interval width	33.64	28.4	31.1	33.74	24.78	25.88	31.68	28.28	28.8	26.84	24.34
Upper confidence interval (Max)	2061	1629	1252	1221	1028	814	806	765	705	637	591
Lower confidence interval (Min)	2027	1601	1220	1187	1003	788	774	736	676	611	567

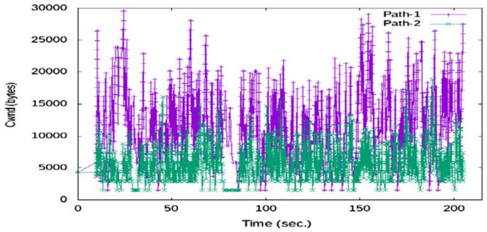
the results plots showing the average throughput performance and the conforming 95% confidence intervals for changing receiver buffer sizes.

Also, we examine the performance of DB-CMT for different receiver buffer sizes concerning time. In this simulation setup, PATH-1 and PATH-2 have a PLR of 1% and 5%. We run two simulations to study the effect of receiver buffer size 32 KB and 64 KB on throughput for 200 s. Figures 15, 16 show that as the buffer size increases, the throughput of all the approaches also increases. Initially, the throughput of all the approaches increases rapidly because of network capacity probing (Slow-Start phase). After the Slow-Start phase, the network identified throughput variations due to packet losses (all mechanisms recover from packet loss by reducing the *cwnd* and retransmission of lost packet). Therefore, we show only the throughput variation part of CMT variants in Figs. 15, 16. CMT and CMT-PF show lower throughput because they cannot competently tolerate packet losses. However, CMT-QA tries to manage packet losses more effectively. As a result, CMT-QA offers better throughput than CMT and CMT-PF. But, DB-CMT uses a delay-based fast retransmission policy to adjust *cwnd* more effectively to improve the available bandwidth utilization. Thus, DB-CMT achieves better throughput than CMT, CMT-PF, and CMT-QA.

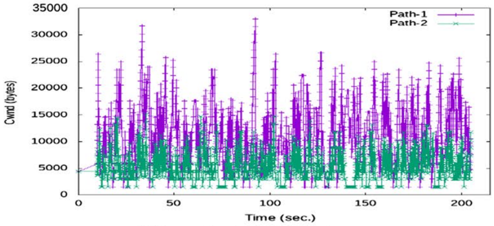
Next, we examine the performance of DB-CMT, CMT, CMT-PF, and CMT-QA in terms of average FTT in lossy and lossless network environments. Figures 17, 18, 19 presents the average FTT performances of all the simulated CMT variants on symmetric and asymmetric PLR when the file size varies from 10 to 90 MB. Both paths have background traffic of 150 kbps. The rest of the simulation configuration remains the same, as shown in Tables 3 and 4 and Fig. 2. First, we examine the performance of DB-CMT in a lossless network environment. Figure 17 shows the average FTT performance of the CMT variants when both paths have a PLR of 0%. The results clearly signify that as the file size increases, FTT increases as well. Here, CMT and CMT-PF offer comparable average FTT performances (585.79 s and 584.37 s) because both approaches use RR scheduling to transmit data over multiple paths. However, CMT-QA transmits data on each path according to their path quality. Thus, CMT-QA offers 1.20% and 1.03% less average FTT performance than CMT and CMT-PF. Further, DB-CMT estimates the load of each path and schedules the data according to path load variation. Therefore, DB-CMT schedules more data on the least loaded path and less on the heavily loaded path. As a result, DB-CMT takes less FTT compared to CMT, CMT-PF, and CMT-QA. In fact, DB-CMT offers 2.80%, 2.63%, and 1.75% lesser FTT than CMT, CMT-PF, and CMT-QA, respectively.

Figure 18 shows the average FTT performance of all the simulated CMT variants when both paths have an asymmetric PLR. In this simulation setup, PATH-1 has a 1% PLR while PATH-2 has a 2% PLR. Figure 18 shows that as file size increases, the FTT of all CMT variants increases as well. However, the difference in FTT between all CMT variants also increases due to a dissimilar PLR of each path. Figure 18 shows that CMT and CMT-PF take similar time due to their similar packet scheduling policy over the multiple available paths. However, CMT-QA takes less time than CMT and CMT-PF because it uses a data distribution policy based on path quality. On the other hand, DB-CMT achieves better performance as compared to CMT, CMT-PF, and CMT-QA due to its delay-based data chunk scheduling policy.

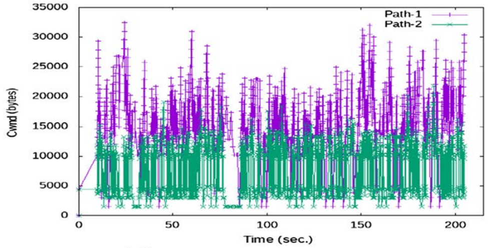
Fig. 8 **A–E** Variations in *cwnd* growth in simulated CMT schemes



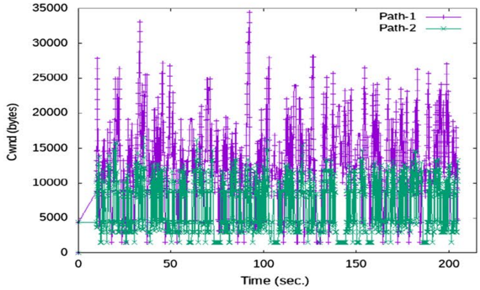
(A) Variations in *cwnd* growth in CMT



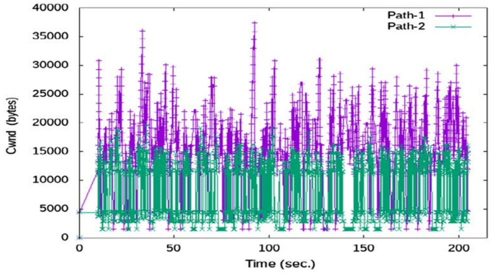
(B) Variations in *cwnd* growth in CMT-PF



(C) Variations in *cwnd* growth in CMT-QA



(D) Variations in *cwnd* growth in A-CMT



(E) Variations in *cwnd* growth in DB-CMT

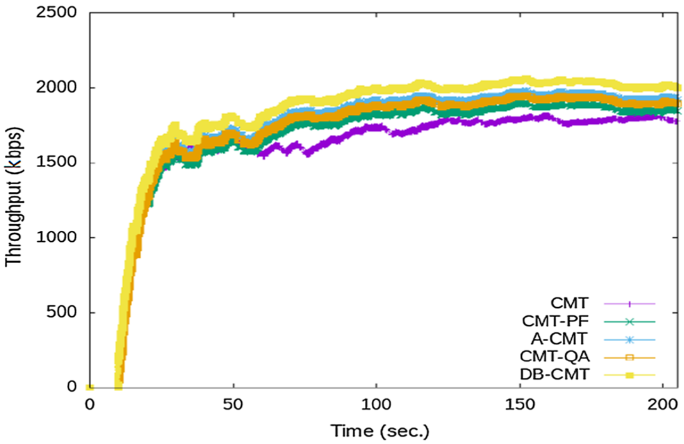


Fig. 9 Throughput (kbps) comparison between simulated CMT schemes

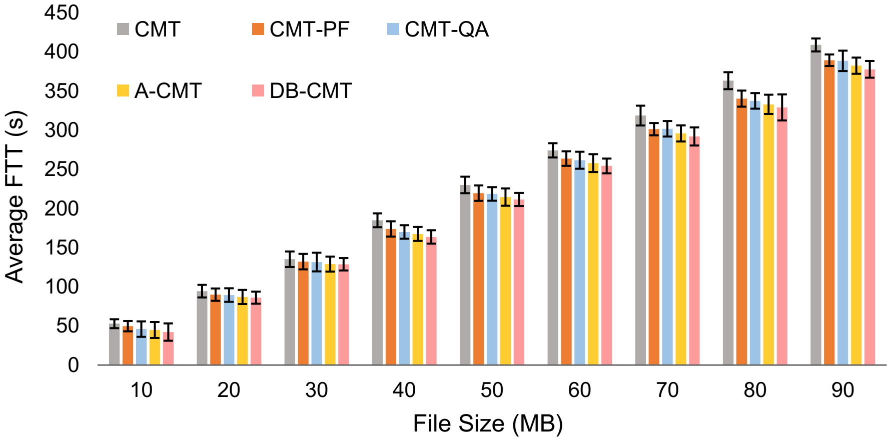


Fig. 10 Average FTT (s) vs. File size of all the simulated CMT variants with 95% confidence interval when an asymmetric loss rate exists

Thus, DB-CMT takes less time to transfer all file sizes in a dissimilar PLR scenario. In the case of symmetric PLR paths, the DB-CMT's FTT performance difference was relatively small compared to other approaches. Still, as the substantive difference in the PLRs of the paths came in, this difference proved to be moderately large. DB-CMT suggests 5.61%, 6.06%, and 5% lesser average FTT performance than CMT, CMT-PF, and CMT-QA, respectively. This proves that DB-CMT is capable of handling the asymmetric PLRs of multiple paths to a larger extent. Figure 19 shows the average FTT performance of all the simulated CMT variants with a 95% confidence interval on symmetric and asymmetric PLR cases (Table 9).

Table 7 Analysis of the Evaluated Schemes via some Statistical Parameters on the basis of Average FTT (s)

Metrics/statistical parameters	Evaluated schemes								
File size (MB)	10	20	30	40	50	60	70	80	90
CMT									
Average FTT (s) after 20 iterations	52.90	94.45	135.20	184.95	230.05	274.25	318.70	363.20	408.85
Sample standard deviation	12.17	17.21	21.18	18.88	22.55	19.42	26.99	23.31	17.74
Margin of error	5.69	8.05	9.91	8.84	10.55	9.09	12.63	10.91	8.3
Confidence interval width	11.38	16.1	19.82	17.68	21.1	18.18	25.26	21.82	16.6
Upper confidence interval (Max)	58.59	102.5	145.11	193.79	240.6	283.34	331.33	374.11	417.15
Lower confidence interval (Min)	47.21	86.4	125.29	176.11	219.5	265.16	306.07	352.29	400.55
CMT-PF									
Average FTT (s) after 20 iterations	49.8	89.9	132.1	173.9	219.55	263.75	301.3	340.25	389.35
Sample standard deviation	14.11	16.86	21.1	20.99	21.09	19.98	16.65	21.95	15.89
Margin of error	6.61	7.89	9.87	9.82	9.87	9.35	7.79	10.27	7.44
Confidence interval width	13.22	15.78	19.74	19.64	19.74	18.7	15.58	20.54	14.88
Upper confidence interval (Max)	56.41	97.79	141.97	183.72	229.42	273.1	309.09	350.52	396.79
Lower confidence interval (Min)	43.19	82.01	122.23	164.08	209.68	254.4	293.51	329.98	381.91
CMT-QA									
Average FTT (s) after 20 iterations	46.00	89.4	131.6	169.95	218.55	261.6	301.75	337.35	388.5
Sample standard deviation	21.10	18.55	25.48	18.53	18.46	23.12	21.26	21.31	27.68
Margin of error	9.88	8.68	11.93	8.67	8.64	10.82	9.95	9.97	12.96
Confidence interval width	19.76	17.36	23.86	17.34	17.28	21.64	19.9	19.94	25.92
Upper confidence interval (Max)	55.88	98.08	143.53	178.62	227.19	272.42	311.7	347.32	401.46
Lower confidence interval (Min)	36.12	80.72	119.67	161.28	209.91	250.78	291.8	327.38	375.54
A-CMT									
Average FTT (s) after 20 iterations	44.9	87.05	128.95	167.5	214.6	257.95	295.85	332.9	382.35
Sample standard deviation	21.79	19.29	20.42	19.03	23.67	24.37	21.94	26.25	22.21
Margin of error	10.2	9.03	9.56	8.91	11.08	11.41	10.27	12.29	10.4
Confidence interval width	20.4	18.06	19.12	17.82	22.16	22.82	20.54	24.58	20.8
Upper confidence interval (Max)	55.1	96.08	138.51	176.41	225.68	269.36	306.12	345.19	392.75
Lower confidence interval (Min)	34.7	78.02	119.39	158.59	203.52	246.54	285.58	320.61	371.95
DB-CMT									
Average FTT (s) after 20 iterations	42.2	86.05	128.75	163.6	211.45	254.35	292.1	329.1	377.65

Table 7 (continued)

Metrics/statistical parameters	Evaluated schemes								
File size (MB)	10	20	30	40	50	60	70	80	90
Sample standard deviation	23.63	16.5	17	18.37	17.89	20.04	24.7	35.48	22.74
Margin of error	11.06	7.72	7.96	8.6	8.37	9.38	11.56	16.61	10.64
Confidence interval width	22.12	15.44	15.92	17.2	16.74	18.76	23.12	33.22	21.28
Upper confidence interval (Max)	53.26	93.77	136.71	172.2	219.82	263.73	303.66	345.71	388.29
Lower confidence interval (Min)	31.14	78.33	120.79	155	203.08	244.97	280.54	312.49	367.01

5.2.3 Statistical Analysis for the Simulated CMT Variants

In this sub-section, we concisely present the basics of the regression analysis notion. This notion directly deals with discovering the correlation between the independent variable (say ‘x, $x=(x_1, x_2, x_3, \ldots, x_n)$’) and a dependent variable (say ‘y, $y=f(x)$’), which is called predictive or regression model). In fact, this kind of model measures the closeness (strength) of that association, discounting the random and outlier values (noises) and predicting the most exact value of the dependent variable ‘y’ on a value of independent variable ‘x’ with a minimum error [73]. This kind of model is based on the Sum of Squares (SoSs), the best conceivable mathematical means to predict the dispersion of points (data). The ultimate objective of this model is to estimate the minimum possible SoSs and plot a line that comes as close as possible to the simulated data.

Mathematically, a linear regression is stated by Eq. (7).

$$y = b(x) + a + \varepsilon \tag{7}$$

where ‘a’ is stated as Y-intercept, it is the anticipated value (mean) of variable ‘y’ when all ‘x’ variables are zero. ‘b’ is defined as the slope of the regression line, which shows the rate of change in variable ‘y’ as the ‘x’ variable deviates. ‘ε’ is the random error which shows the deviation between the predicted and the actual value of a dependent variable. We have applied the linear regression analysis to predict the next value of throughput for the simulated methods, dependent on a PLR factor. Now, we need to understand and estimate the quality of the regression model applied to our data set. The factors such as *p*-values and R-Square (R^2) are the essential criteria to assess the quality of the regression model. R^2 is the determination coefficient which is basically utilized as an indicator of the goodness of fit. It indicates how many the number of points come over the regression line. Specifically, R^2 can be computed by Eq. (8).

$$\mathrm{R}^2 = 1 - (\text{Sum Squared Regression (SSR)}/\text{Total SoSs (TSoSs)}),$$

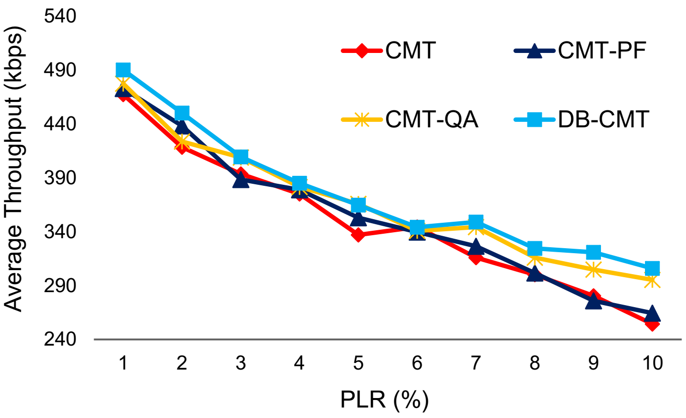


Fig. 11 Average throughput vs. PLR at 32 KB receiver buffer size

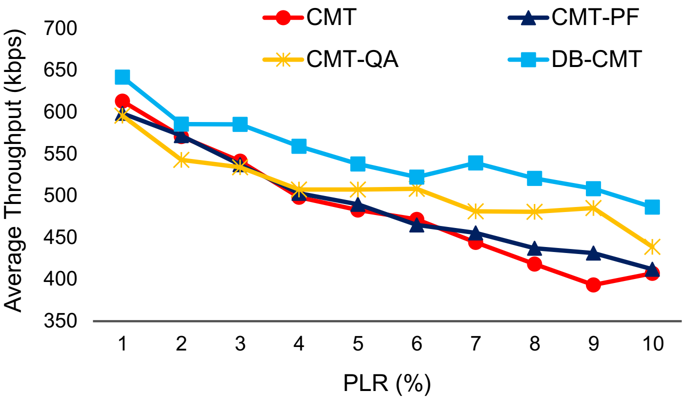


Fig. 12 Average throughput vs. PLR at 64 KB receiver buffer size

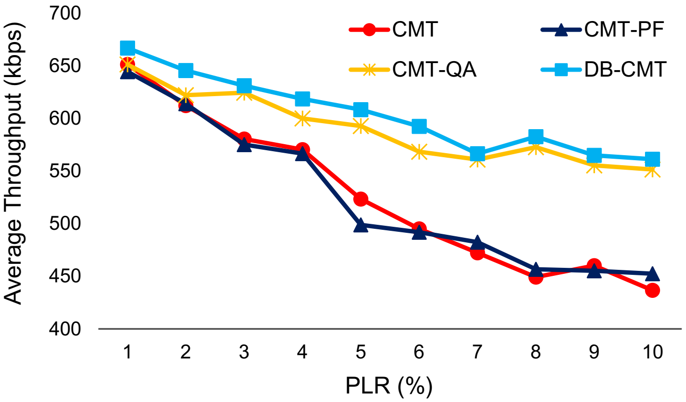


Fig. 13 Average throughput vs. PLR at 128 KB receiver buffer size

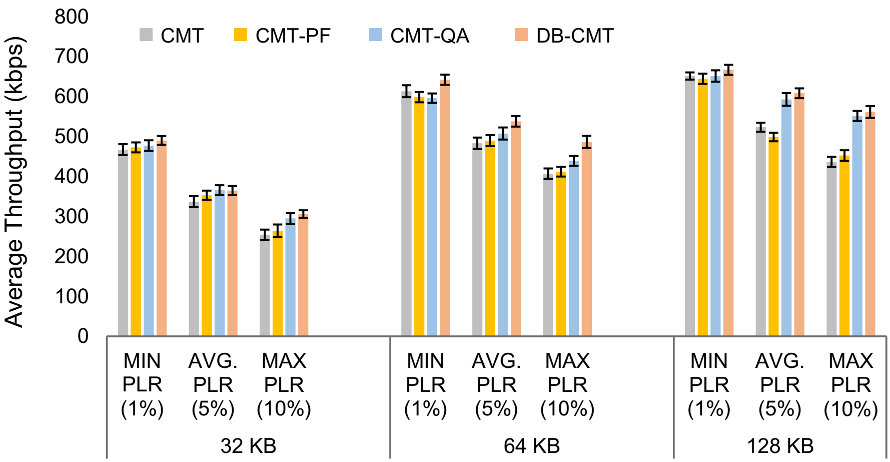


Fig. 14 Average throughput performances of all the simulated CMT variants with 95% confidence interval on varying receiver buffer size considering minimum, average, and maximum PLR

$$\text{with SSR} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \text{ and TSoSs} = \sum_{i=1}^{n} (y_i - \bar{y})^2 \tag{8}$$

Here, '$\bar{y}$' is the mean value of simulated data points (actual y value) 'y_i' and '$\hat{y}_i$'(predicted y value) is calculated by placing 'x_i' value into the considered regression model 'y=f(x)', SSR is the summation of residual (r_i) squared (i.e., $r_i = y_i - \hat{y}_i$) and TSoSs is the summation of the distance the simulated data is far from the mean all squared. The value R^2 criterion ranges from 0 to 1 (i.e., $R^2 \in [0,1]$). Here, in our scenario, after applying such linear regression analysis separately on each simulated CMT variant's data set, we got the regression analysis results shown in Table 10. These results show how well the estimated linear regression equalities (shown in Eqs. (9–13)) fit our simulated data. After applying linear regression to each simulated method, we get the following equations for each method:

$$\text{CMT} : \text{y} = -115.63\text{x} + 1465.1 \tag{9}$$

$$\text{CMT-PF} : \text{y} = -120.7\text{x} + 1470.5 \tag{10}$$

$$\text{CMT-QA} : \text{y} = -122.06\text{x} + 1577.4 \tag{11}$$

$$\text{A-CMT} : \text{y} = -122.86\text{x} + 1557.3 \tag{12}$$

$$\text{DB-CMT} : \text{y} = -127.82\text{x} + 1671 \tag{13}$$

Here, Multiple R is the correlation coefficient and measures the strength of association (linear) between two variables. The value of the Multiple R criterion ranges from

Table 8 Analysis of the Evaluated Schemes via some Statistical Parameters on the basis of Average Throughput (kbps) considering PLR cases on changing receiver buffer sizes

Metrics/statistical parameters	Evaluated schemes								
Receiver buffer size	32 KB			64 KB			128 KB		
Consider PLR Cases	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
CMT									
Average throughput (kbps) after 20 iterations	467.23	336.96	254.27	613.37	483.03	407.24	651.44	523.43	436.67
Sample standard deviation	29.49	29.1	27.47	32.03	30.66	27.46	19.32	23.88	27.19
Margin of error	13.8	13.62	12.86	14.99	14.35	12.85	9.04	11.18	12.72
Confidence interval width	27.6	27.24	25.72	29.98	28.7	25.7	18.08	22.36	25.44
Upper confidence interval (Max)	481.04	350.58	267.12	628.36	497.37	420.09	660.48	534.61	449.4
Lower confidence interval (Min)	453.43	323.35	241.41	598.38	468.68	394.39	642.39	512.25	423.95
Metrics/statistical parameters	Evaluated schemes								
Receiver buffer size	32 KB			64 KB			128 KB		
Consider PLR cases	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
CMT-PF									
Average throughput (kbps) after 20 iterations	472.85	352.93	264.38	598.77	489.82	412.21	644.54	498.94	452.47
Sample standard deviation	26.66	25.4	33.49	27.56	29.59	26.13	27.5	23.01	28.32
Margin of error	12.48	11.89	15.67	12.9	13.85	12.23	12.87	10.77	13.26
Confidence interval width	24.96	23.78	31.34	25.8	27.7	24.46	25.74	21.54	26.52
Upper confidence interval (Max)	485.33	364.81	280.05	611.67	503.67	424.44	657.41	509.71	465.72
Lower confidence interval (Min)	460.37	341.04	248.7	585.87	475.97	399.98	631.67	488.17	439.21

Table 8 (continued)

Metrics/statistical parameters	Evaluated schemes								
Receiver buffer size	32 KB			64 KB			128 KB		
Consider PLR cases	Min	Avg	Max	Min	Avg	Max	Min	Avg	Max
CMT-QA									
Average throughput (kbps) after 20 iterations	477.3	365.72	295.28	595.94	507.55	438.91	651.43	592.87	551.53
Sample standard deviation	28.82	26.32	29.58	25.39	32.19	26.74	30.54	34.01	27.01
Margin of error	13.49	12.32	13.84	11.88	15.07	12.51	14.29	15.92	12.64
Confidence interval width	26.98	24.64	27.68	23.76	30.14	25.02	28.58	31.84	25.28
Upper confidence interval (Max)	490.79	378.04	309.12	607.82	522.61	451.42	665.73	608.79	564.17
Lower confidence interval (Min)	463.82	353.41	281.44	584.05	492.48	426.39	637.14	576.95	538.89
Metrics/statistical parameters	Evaluated schemes								
Receiver buffer size	32 KB			64 KB			128 KB		
Consider PLR cases	MIN	Avg	MAX	MIN	Avg	MAX	MIN	Avg	MAX
DB-CMT									
Average throughput (kbps) after 20 iterations	490.11	364.67	305.89	642.11	538.13	486.48	666.95	608.37	561.27
Sample standard deviation	23.54	24.54	20.19	27.06	28.2	32.57	26.74	26.33	31.67
Margin of error	11.02	11.49	9.45	12.67	13.2	15.24	12.51	12.32	14.82
Confidence interval width	22.04	22.98	18.9	25.34	26.4	30.48	25.02	24.64	29.64
Upper confidence interval (Max)	501.13	376.16	315.34	654.78	551.33	501.73	679.46	620.7	576.1
Lower confidence interval (Min)	479.09	353.19	296.45	629.45	524.93	471.24	654.43	596.05	546.45

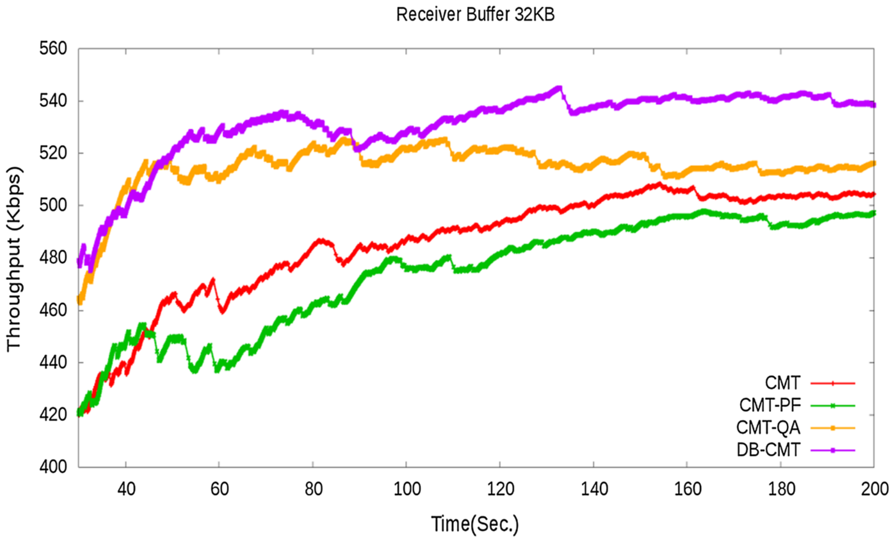


Fig. 15 Throughput vs. simulation time at 32 KB receiver buffer

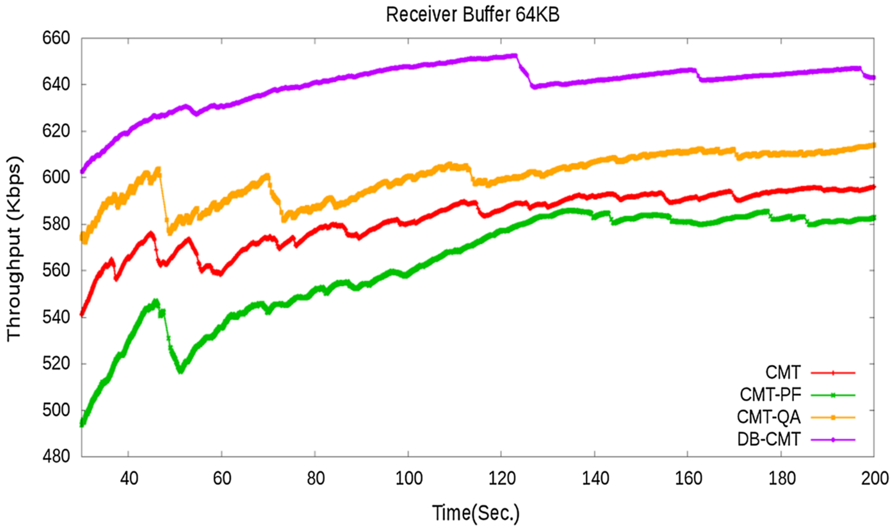


Fig. 16 Throughput vs. simulation time at 64 KB receiver buffer

-1 to 1, and its absolute value shows the strength of the association. The more will be the absolute value, the stronger the association. If its value reaches close to 1, then there is a strong positive association. Here, from the results shown in Table 10, it can be perceived that in all the simulated CMT variant cases, the value of the Multiple R criterion is close to 1; hence, there is a strong positive relationship between the dependent variable (throughput) and the independent variable (PLR). Apart from this, it can

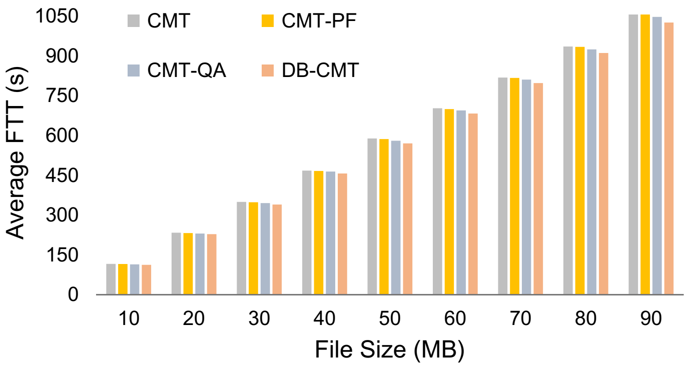


Fig. 17 Average FTT vs. file size when a symmetric loss rate exists

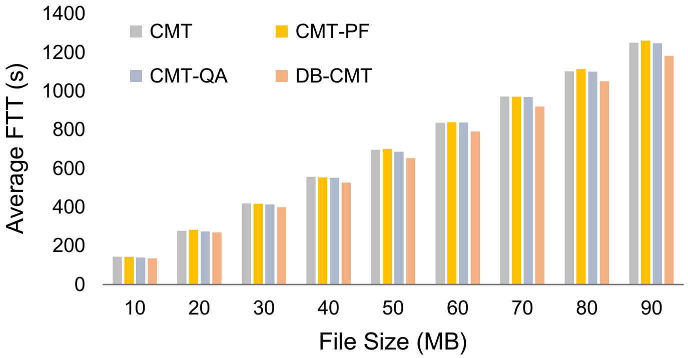


Fig. 18 Average FTT vs. file size when an asymmetric loss rate exists

also be understood from the results shown in Table 10 that the R^2 criterion amongst all the CMT variants is close to 0.85. It means that 85% of our simulated data values fit the regression analysis model. Moreover, 85% of our considered dependent variable (throughput) are explicated by the independent variable (PLR). To visualize the association between the two considered variables, we plot the results, which are shown in Fig. 20A–E.

6 Summary of Assessed Simulation Results

This section summarizes the complete impact of our suggested scheme (DB-CMT) in a multipath environment. We compared DB-CMT with the conventional RR-based data scheduling scheme, i.e., CMT. Moreover, we compared and analyzed DB-CMT against adaptive versions of CMT such as CMT-PF, CMT-QA, and A-CMT in an environment where bandwidth is sufficiently available. First, we compared the throughput performance of DB-CMT with CMT, CMT-PF, A-CMT, and

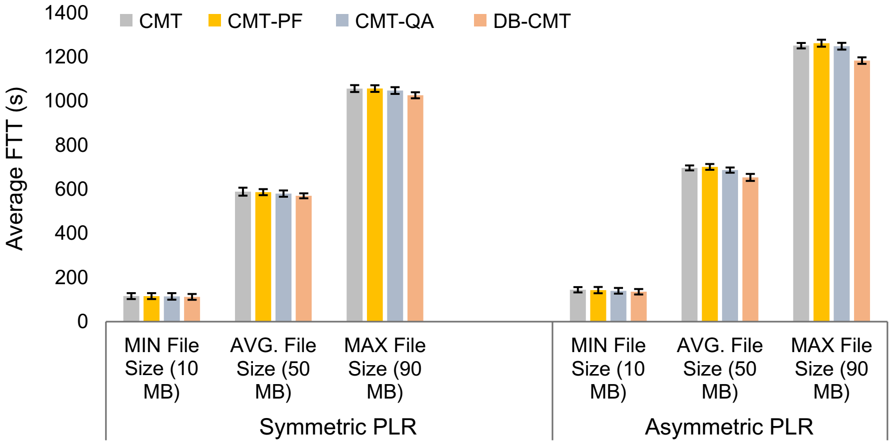


Fig. 19 Average FTT performances of all the simulated CMT variants with 95% confidence interval on symmetric/asymmetric PLRs considering minimum, average, and maximum file sizes

CMT-QA when the PLR varies between 0 to 10% with a 64 KB receiver buffer size (default). Then, considering the same environment, we compared the *cwnd* growth variations with respect to the simulated CMT schemes. By performing a thorough analysis of throughput and *cwnd* growth variations, it has been confirmed that DB-CMT tolerated packet loss better and utilized the obtainable aggregate bandwidth competently than other simulated CMT schemes. Afterward, in the same network environment, we compared and analyzed the FTT performance on asymmetric loss rate situations with respect to the simulated CMT schemes. We thoroughly analyzed those FTT results and found that DB-CMT takes the least FTT to transmit each size of the file compared to other simulated CMT schemes.

Furthermore, we also compared and analyzed DB-CMT against adaptive versions of CMT in an environment where bandwidth is extremely limited. We compared and analyzed the throughput performance of DB-CMT with CMT, CMT-PF, and CMT-QA when the PLR varies between 1 to 10%. For performing analysis and comparisons, we run a group of three types of simulations (i.e., for the receiver buffer sizes of 32 KB, 64 KB, and 128 KB) to understand the impact of different receiver buffer sizes on the throughput performance of the simulated schemes. It has been perceived that the throughput performance of all simulated schemes surges with an upsurge in the sizes of the receiver buffer. The lower the buffer size is, the greater the possibilities are (1) of triggering the receiver buffer blocking issue, (2) of transmitting NACKs, and (3) of reducing *cwnd* size by the sender. Consequently, such a situation leads to the problem of lower channel utilization, ultimately leading to inferior throughput performance. This simulation study validates that the receiver buffer size affects the throughput performance of all the considered CMT variants. Comparing DB-CMT with the rest of the simulated CMT schemes, it is clear that the DB-CMT scheme is better at utilizing the obtainable aggregated bandwidth from different links in this environment.

Table 9 Analysis of the evaluated schemes via some statistical parameters on the basis of average FTT (s) considering symmetric/asymmetric PLR cases

Metrics/statistical parameters	Evaluated schemes					
Considered PLR cases	Symmetric PLR			Asymmetric PLR		
Considered file sizes	Min	Avg	Max	Min	Avg	Max
CMT						
Average FTT (s) after 20 iterations	116.1	589.39	1056.66	144.66	696.87	1251.15
Sample standard deviation	28.92	38.99	33.22	25.48	24.31	26.57
Margin of error	13.54	18.25	15.55	11.93	11.38	12.44
Confidence interval width	27.08	36.5	31.1	23.86	22.76	24.88
Upper confidence interval (Max)	129.64	607.64	1072.2	156.59	708.25	1263.59
Lower confidence interval (Min)	102.57	571.14	1041.11	132.74	685.5	1238.72
CMT-PF						
Average FTT (s) after 20 iterations	115.95	587.01	1056.71	142.99	701.56	1262.11
Sample standard deviation	28.65	29.49	32.55	30.71	27.57	33.74
Margin of error	13.41	13.8	15.23	14.37	12.9	15.79
Confidence interval width	26.82	27.6	30.46	28.74	25.8	31.58
Upper confidence interval (Max)	129.36	600.81	1071.94	157.36	714.46	1277.91
Lower confidence interval (Min)	102.55	573.2	1041.48	128.61	688.66	1246.32
CMT-QA						
Average FTT (s) after 20 iterations	114.73	580.54	1047.76	140.06	687.12	1248.56
Sample standard deviation	31.72	30.1	32.71	27.67	24.79	33
Margin of error	14.85	14.09	15.31	12.95	11.6	15.44
Confidence interval width	29.7	28.18	30.62	25.9	23.2	30.88
Upper confidence interval (Max)	129.57	594.63	1063.07	153.01	698.72	1264
Lower confidence interval (Min)	99.88	566.45	1032.45	127.11	675.52	1233.11
DB-CMT						
Average FTT (s) after 20 iterations	112.64	570.64	1026.25	135.8	653.97	1183.37
Sample standard deviation	28.06	23.84	29.04	26.17	33.45	31.97
Margin of error	13.13	11.16	13.59	12.25	15.66	14.96
Confidence interval width	26.26	22.32	27.18	24.5	31.32	29.92
Upper confidence interval (Max)	125.77	581.79	1039.84	148.05	669.63	1198.33
Lower confidence interval (Min)	99.51	559.48	1012.66	123.56	638.32	1168.41

Then, we further assessed the average FTT performances with varying file sizes in symmetric and asymmetric loss rate situations. We compared the FTT performances for DB-CMT, CMT, CMT-PF, A-CMT, and CMT-QA. DB-CMT utilizes a new delay-based data chunk scheduling, a new retransmission path selection policy (RTX-CL), and a fast retransmission policy. DB-CMT estimates the transmission load of each path and schedules the data according to the path load variation. Consequently, DB-CMT can schedule more data on the least loaded path and less on the severely burdened path. For this reason, DB-CMT achieves more throughput and less average FTT performances.

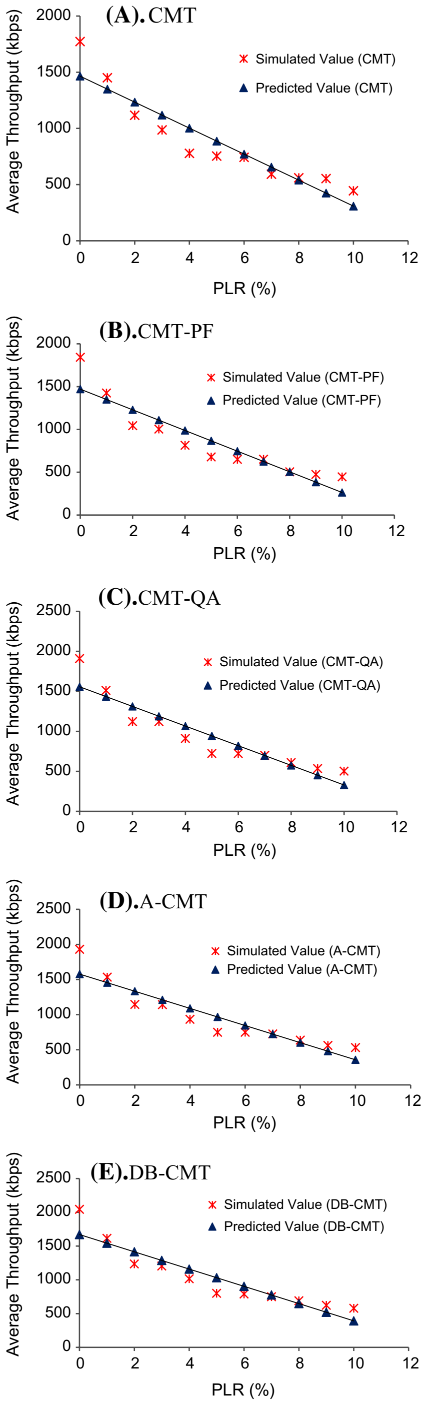
Table 10 Regression statistics for simulated CMT variants

Criteria	Simulated CMT variants				
	CMT	CMT-PF	CMT-QA	A-CMT	DB-CMT
Multiple R	0.926077	0.916317	0.922453	0.921586	0.921735
R^2	0.857618	0.839637	0.850919	0.84932	0.849596
Standard error	164.7129	184.4142	179.7856	179.7345	188.0205

Based on the simulation results obtained, the following statements can be claimed:

1. The proposed delay-based data chunk scheduling scheme for CMT can transmit more data chunks through the minimum delay variation path that subsequently minimizes the out-of-order data chunk delivery and further minimizes the possibility of receiver buffer blocking. Hence, it later reduces the likelihood of *cwnd* growth reductions and further assists DB-CMT in maintaining the better utilization of the obtainable aggregated bandwidth from different links.
2. In the DB-CMT scheme, the source approximates an expected and the actual transmission rate for each destination. Based on these estimations, DB-CMT efficiently assesses the load on a path. This technique can control the aggressiveness of *cwnd* growth and subsequently offers adequate time to settle the congestion condition in the network.
3. DB-CMT comprises a new retransmission path selection policy (RTX-CL) which chooses the best retransmission path according to the highest *cwnd* and lower loss rate. This scheme effectively eliminates the issue of randomness in the path selection process if numerous paths have identical *cwnd* or *ssthresh* factors.
4. DB-CMT also includes a delay-based fast retransmission policy which eradicates the problem of the conventional fast retransmission policy. Specifically, due to the four SACKs receptions, the traditional fast retransmission policy blindly reduces the *cwnd* size to half. But, SACKs can also be received when the receiver gets unordered data chunks. Hence, such frequent reduction in *cwnd* size leads to the issue of compromised throughput.
5. DB-CMT scheme can considerably improve the throughput performance on varying PLRs with varying receiver buffer sizes by transmitting more data chunks through a minimum delay variation path. Moreover, DB-CMT can also improve the FTT performance on varying file sizes compared to other simulated CMT schemes.

Fig. 20 **A**–**E**: Regression results for the simulated CMT variants



7 Conclusions

This paper proposed a new delay-based concurrent multipath transfer policy (DB-CMT) for simultaneous multipath transmission. DB-CMT has a new delay-based data chunk scheduling policy, a new retransmission path selection policy (RTX-CL), and a fast retransmission method. The proposed data chunk scheduling policy transmits a large amount of data over a less loaded path, while RTX-CL selects the retransmission path whose *cwnd* is larger and the PLR is lower. Simulation results showed that the proposed RTX-CL policy achieves better throughput and retransmission timeout results. Moreover, the overall performance of DB-CMT is better regarding throughput, FTT, and *cwnd* growth.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions.

Declarations

Competing interest The authors declare no conflicts of interest.

References

1. Li, M., Lukyanenko, A., Ou, Z., Ylä-Jääski, A., Tarkoma, S., Coudron, M., Secci, S.: Multipath transmission for the internet: a survey. IEEE Commun. Surv. Tutor. **18**(4), 2887–2925 (2016)
2. Postel, J.: Transmission Control Protocol. IETF Internet-Draft, September 1981. (1981). https://tools.ietf.org/html/rfc793, Accessed 20 Sept 2020.
3. Postel, J.: User Datagram Protocol, IETF Internet-Draft, August 1980. (1980). https://tools.ietf.org/html/rfc768, Accessed 19 Apr 2021.
4. Habib, S., Qadir, J., Ali, A., Habib, D., Li, M., Sathiaseelan, A.: The past, present, and future of transport-layer multipath. J. Netw. Comput. Appl. **75**, 236–258 (2016)
5. Yaqoob, A., Bi, T., Muntean, G.M.: A survey on adaptive 360 video streaming: solutions, challenges and opportunities. IEEE Commun. Surv. Tutor. **22**(4), 2801–2838 (2020)
6. Pan, S., Zhou, Y., Zhang, Z., Yang, S., Qian, F., Hu, G.: Identify congested links with network tomography under multipath routing. J. Netw. Syst. Manage. **27**(2), 409–429 (2019)
7. Sharma, V.K., Verma, L.P., Kumar, M., Naha, R.K., Mahanti, A.: A-CAFDSP: an adaptive-congestion aware Fibonacci sequence based data scheduling policy. Comput. Commun. **158**, 141–165 (2020)
8. Verma, L.P., Sharma, V.K., Kumar, M., Mahanti, A.: An adaptive multi-path data transfer approach for MP-TCP. Wirel. Netw. (2022). https://doi.org/10.1007/s11276-022-02958-2
9. Verma, L.P., Sharma, V.K., Kumar, M., Kanellopoulos, D.: A novel delay-based adaptive congestion control TCP variant. Comput. Electr. Eng. **101**, 108076 (2022)

10. Almadani, B., Alsaeedi, M., Al-Roubaiey, A.: QoS-aware scalable video streaming using data distribution service. Multimed. Tools Appl. **75**(10), 5841–5870 (2016)
11. Barakabitze, A.A., Barman, N., Ahmad, A., Zadtootaghaj, S., Sun, L., Martini, M.G., Atzori, L.: QoE management of multimedia streaming services in future networks: a tutorial and survey. IEEE Commun. Surv. Tutor. **22**(1), 526–565 (2019)
12. Xu, C., Zhao, J., Muntean, G.M.: Congestion control design for multipath transport protocols: a survey. IEEE Commun. Surv. Tutor. **18**(4), 2948–2969 (2016)
13. Xu, C., Zhao, F., Guan, J., Zhang, H., Muntean, G.M.: QoE-driven user-centric VoD services in urban multihomed P2P-based vehicular networks. IEEE Trans. Veh. Technol. **62**(5), 2273–2289 (2012)
14. Xu, C., Jia, S., Wang, M., Zhong, L., Zhang, H., Muntean, G.M.: Performance-aware mobile community-based VoD streaming over vehicular Ad hoc networks. IEEE Trans. Veh. Technol. **64**(3), 1201–1217 (2014)
15. Xu, C., Li, Z., Li, J., Zhang, H., Muntean, G.M.: Cross-layer fairness-driven concurrent multipath video delivery over heterogeneous wireless networks. IEEE Trans. Circuits Syst. Video Technol. **25**(7), 1175–1189 (2014)
16. Xu, C., Jia, S., Zhong, L., Zhang, H., Muntean, G.M.: Ant-Inspired mini-community-based solution for video-on-demand services in wireless mobile networks. IEEE Trans. Broadcast. **60**(2), 322–335 (2014)
17. Cao, Y., Chen, J., Liu, Q., Lei, G., Wang, H., You, I.: Can multipath TCP be robust to cyber attacks with incomplete information? IEEE Access **8**, 165872–165883 (2020)
18. Stewart, R.: Stream control transmission protocol. IETF Internet-Draft, September 2007. (2007). https://tools.ietf.org/html/rfc4960. Accessed 24 Feb 2021
19. Iyengar, J.R., Amer, P.D., Stewart, R.: Concurrent multipath transfer using SCTP multihoming over independent end-to-end paths. IEEE/ACM Trans. Network. **14**(5), 951–964 (2006)
20. Paasch, C., Bonaventure, O.: Multipath TCP. Commun. ACM **57**(4), 51–57 (2014)
21. Raiciu, C., Handly, M., Wischik, D.: Coupled congestion control for multipath transport protocols. IETF Internet-Draft, October 2011. (2011). https://tools.ietf.org/html/rfc6356. Accessed 29 Mar 2019.
22. Ford, A., Raiciu, C., Handly, M., Bonaventure, O. TCP Extensions for Multipath Operation with Multiple Addresses. IETF Internet-Draft, January 2013. (2013). https://tools.ietf.org/html/rfc6824. Accessed 19 Mar 2019
23. Bonaventure, O.: In Korean, multipath TCP is pronounced GIGA path. (2015). http://blog.multipath-tcp.org/blog/html/2015/07/24/korea.html. Accessed 10 Mar 2020
24. Xu, C., Wang, P., Xiong, C., Wei, X., Muntean, G.M.: Pipeline network coding-based multipath data transfer in heterogeneous wireless networks. IEEE Trans. Broadcast. **63**(2), 376–390 (2017)
25. OVH Company.: Overthebox. (2015). https://www.ovhtelecom.fr/overthebox. Accessed 12 Feb 2020.
26. Hybrid Internet Access Bonding. (2020). http://www.tessares.net. Accessed 12 Apr 2021.
27. Wallace, T.D., Shami, A.: A review of multihoming issues using the stream control transmission protocol. IEEE Commun. Surv. Tutor. **14**(2), 565–578 (2011)
28. Verma, L.P., Sharma, V.K., Kumar, M.: New delay-based fast retransmission policy for CMT-SCTP. Int. J. Intell. Syst. Appl **10**(3), 59–66 (2018)
29. Sharma, V.K., Verma, L.P., Kumar, M.: CL-ADSP: cross-layer adaptive data scheduling policy in mobile Ad-hoc networks. Futur. Gener. Comput. Syst. **97**, 530–563 (2019)
30. Yang, T., Pan, L., Jian, L., Hongcheng, H., Jun, W.: Reducing receive buffer blocking in CMT based on SCTP using retransmission policy. In 2011 IEEE 3rd International Conference on Communication Software and Networks (pp. 122–125). IEEE (2011)
31. Singh, S.K., Das, T., Jukan, A.: A survey on internet multipath routing and provisioning. IEEE Commun. Surv. Tutor. **17**(4), 2157–2175 (2015)
32. Aschenbrenner, F., Shreedhar, T., Gasser, O., Mohan, N., Ott, J. From single lane to highways: analyzing the adoption of multipath TCP in the internet. In: *IFIP networking conference (networking)*. (2021)
33. Dreibholz, T., Becke, M., Rathgeb, E. P., Tüxen, M.: On the use of concurrent multipath transfer over asymmetric paths. In 2010 IEEE Global Telecommunications Conference GLOBECOM 2010 (pp. 1–6). IEEE (2010)
34. Dreibholz, T., Rathgeb, E.P., Rüngeler, I., Seggelmann, R., Tüxen, M., Stewart, R.R.: Stream control transmission protocol: past, current, and future standardization activities. IEEE Commun. Mag. **49**(4), 82–88 (2011)

35. Natarajan, P., Ekiz, N., Amer, P.D., Stewart, R.: concurrent multipath transfer during path failure. Comput. Commun. **32**(15), 1577–1587 (2009)
36. Yilmaz, E., Ekiz, N., Natarajan, P., Amer, P.D., Leighton, J.T., Baker, F., Stewart, R.R.: Throughput analysis of non-renegable selective acknowledgments (NR-SACKs) for SCTP. Comput. Commun. **33**(16), 1982–1991 (2010)
37. Xu, C., Liu, T., Guan, J., Zhang, H., Muntean, G.M.: CMT-QA: quality-aware adaptive concurrent multipath data transfer in heterogeneous wireless networks. IEEE Trans. Mob. Comput. **12**(11), 2193–2205 (2012)
38. Xu, C., Li, Z., Zhong, L., Zhang, H., Muntean, G.M.: CMT-NC: improving the concurrent multipath transfer performance using network coding in wireless networks. IEEE Trans. Veh. Technol. **65**(3), 1735–1751 (2015)
39. Shailendra, S., Bhattacharjee, R., Bose, S.K.: MPSCTP: a simple and efficient multipath algorithm for SCTP. IEEE Commun. Lett. **15**(10), 1139–1141 (2011)
40. Shailendra, S., Bhattacharjee, R., Bose, S.K.: An implementation of min-max optimization for multipath SCTP through bandwidth estimation based resource pooling technique. AEU-Int. J. Electron. Commun. **67**(3), 246–249 (2013)
41. Shailendra, S., Bhattacharjee, R., Bose, S.K.: A multipath variant of SCTP with optimized flow division extension. Comput. Commun. **67**, 56–65 (2015)
42. Hwang, Y., Saha, A., Choi, H., Lim, H., Obele, B.O.: HMTP: multipath transport protocol for multihoming wireless erasure networks. Trans. Emerg. Telecommun. Technol. **26**(8), 1061–1072 (2015)
43. Verma, L.P., Sheel, N., Yadev, C.S.: Concurrent multipath transfer using delay aware scheduling. In: Innovations in Computational Intelligence and Computer Vision, pp. 247–255. Springer, Singapore (2021)
44. Wu, J., Cheng, B., Yuen, C., Shang, Y., Chen, J.: Distortion-aware concurrent multipath transfer for mobile video streaming in heterogeneous wireless networks. IEEE Trans. Mob. Comput. **14**(4), 688–701 (2014)
45. Wu, J., Yuen, C., Wang, M., Chen, J.: Content-aware concurrent multipath transfer for high-definition video streaming over heterogeneous wireless networks. IEEE Trans. Parallel Distrib. Syst. **27**(3), 710–723 (2016)
46. Wu, J., Cheng, B., Wang, M.: Improving multipath video transmission with raptor codes in heterogeneous wireless networks. IEEE Trans. Multimed. **20**(2), 457–472 (2018)
47. Chen, H., Zhang, X., Xu, Y., Ma, Z., Zhang, W.: Efficient mobile video streaming via context-aware raptorq-based unequal error protection. IEEE Trans. Multimed. **22**(2), 459–473 (2019)
48. Hellge, C., Gómez-Barquero, D., Schierl, T., Wiegand, T.: Layer-aware forward error correction for mobile broadcast of layered media. IEEE Trans. Multimed. **13**(3), 551–562 (2011)
49. Chen, F., Zhang, J., Zheng, M., Wu, J., Ling, N.: Long-term rate control for concurrent multipath real-time video transmission in heterogeneous wireless networks. J. Visual Commun. Image Represent. **77**, 102999 (2021)
50. Wu, J., Cheng, B., Wang, M., Chen, J.: Priority-aware FEC coding for high-definition mobile video delivery using TCP. IEEE Trans. Mob. Comput. **16**(4), 1090–1106 (2016)
51. Wu, J., Yuen, C., Cheng, B., Wang, M., Chen, J.: Energy-minimized multipath video transport to mobile devices in heterogeneous wireless networks. IEEE J. Sel. Areas Commun. **34**(5), 1160–1178 (2016)
52. Wu, J., Cheng, B., Wang, M., Chen, J.: Energy-aware concurrent multipath transfer for real-time video streaming over heterogeneous wireless networks. IEEE Trans. Circ. Syst. Video Technol. **28**(8), 2007–2023 (2017)
53. Yedugundla, K., Ferlin, S., Dreibholz, T., Alay, Ö., Kuhn, N., Hurtig, P., Brunstrom, A.: Is multipath transport suitable for latency sensitive traffic? Comput. Netw. **105**, 1–21 (2016)
54. Li, M., Lukyanenko, A., Cui, Y.: Network coding based multipath TCP. In: 2012 proceedings *IEEE INFOCOM workshops* (pp. 25–30). IEEE (2012)
55. Li, M., Lukyanenko, A., Tarkoma, S., Cui, Y., Ylä-Jääski, A.: Tolerating path heterogeneity in multipath TCP with bounded receive buffers. Comput. Netw. **64**, 1–14 (2014)
56. Cui, Y., Wang, L., Wang, X., Wang, H., Wang, Y.: FMTCP: a fountain code-based multipath transmission control protocol. IEEE/ACM Trans. Netw. **23**(2), 465–478 (2014)
57. Zhou, D., Song, W., Shi, M.: Goodput improvement for multipath TCP by Congestion window adaptation in multi-radio devices. In: 2013 IEEE 10th Consumer Communications and Networking Conference (CCNC), pp. 508–514. IEEE (2013)

58. Wu, J., Yuen, C., Cheng, B., Yang, Y., Wang, M., Chen, J.: Bandwidth-efficient multipath transport protocol for quality-guaranteed real-time video over heterogeneous wireless networks. IEEE Trans. Commun. **64**(6), 2477–2493 (2016)
59. Paasch, C., Ferlin, S., Alay, O., Bonaventure, O.: Experimental evaluation of multipath TCP schedulers. In: Proceedings of the 2014 ACM SIGCOMM workshop on capacity sharing workshop (pp. 27–32) (2014)
60. Ferlin, S., Alay, Ö., Mehani, O., Boreli, R.: BLEST: blocking estimation-based MPTCP scheduler for heterogeneous networks. In: 2016 IFIP Networking Conference (IFIP Networking) and Workshops, pp. 431–439. IEEE (2016)
61. Raiciu, C., Paasch, C., Barre, S., Ford, A., Honda, M., Duchene, F., et al.: How hard can it be? Designing and implementing a deployable multipath {TCP}. In: 9th {USENIX} symposium on networked systems design and implementation ({NSDI} 12), pp. 399–412. (2012)
62. Ferlin, S., Dreibholz, T., Alay, Ö.: Multi-path transport over heterogeneous wireless networks: does it really pay off? In: 2014 IEEE global communications conference, pp. 4807–4813. IEEE (2014)
63. Ferlin, S., Alay, Ö., Dreibholz, T., Hayes, D.A., Welzl, M.: Revisiting congestion control for multipath TCP with shared bottleneck detection. In: IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications, pp. 1–9. IEEE (2016)
64. Lim, Y.S., Nahum, E.M., Towsley, D., Gibbens, R.J.: ECF: an MPTCP path scheduler to manage heterogeneous paths. In: Proceedings of the 13th International Conference on Emerging Networking Experiments and Technologies, pp. 147–159. (2017)
65. Shi, H., Cui, Y., Wang, X., Hu, Y., Dai, M., Wang, F., Zheng, K.: {STMS}: improving {MPTCP} throughput under heterogeneous networks. In: 2018 {USENIX} Annual Technical Conference ({USENIX}{ATC} 18), pp. 719–730. (2018)
66. Wu, H., Alay, Ö., Brunstrom, A., Ferlin, S., Caso, G.: Peekaboo: learning-based multipath scheduling for dynamic heterogeneous environments. IEEE J. Sel. Areas Commun. **38**(10), 2295–2310 (2020)
67. Hurtig, P., Grinnemo, K.J., Brunstrom, A., Ferlin, S., Alay, Ö., Kuhn, N.: Low-latency scheduling in MPTCP. IEEE/ACM Trans. Networking **27**(1), 302–315 (2018)
68. Hayes, D., et al.: Report on prototype development and evaluation of end-system, application layer and API mechanisms. Simula Res. Lab., Oslo, Norway, Tech. Rep. RITE EU FP7-ICT, Sep. 2015. (2015). https://riteproject.files.wordpress.com/2015/12/rite_deliverable_1-3.pdf. Accessed 23 Apr 2020
69. Han, J., Xing, Y., Xue, K., Wei, D.S., Xue, G., Hong, P.: Leveraging coupled BBR and adaptive packet scheduling to boost MPTCP. *arXiv preprint* arXiv:2002.06284. *(2020)*
70. Gao, W., Huang, J., Zou, S., Li, W., Wang, J., Chen, J.: AAC: adaptively adjusting concurrency by exploiting path diversity in datacenter networks. J. Netw. Syst. Manage. **29**(3), 1–26 (2021)
71. Verma, L.P., Kumar, M.: An adaptive data chunk scheduling for concurrent multipath transfer. Comput. Stand. Interfaces **52**, 97–104 (2017)
72. The Network Simulator-ns-2. http://www.isi.edu/nsnam/ns. Accessed 20 Sept 2019.
73. Tan, Q., Yang, X., Zhao, L., Zhou, X., Dreibholz, T.: A statistic procedure to find formulae for buffer size in MPTCP. In: 2018 IEEE 3rd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), pp. 900–907. IEEE (2018)
74. Liu, S., Lei, W., Zhang, W., Guan, Y.: CMT-SR: a selective retransmission based concurrent multipath transmission mechanism for conversational video. Comput. Netw. **112**, 360–371 (2017)
75. Arianpoo, N., Aydin, I., Leung, V.C.: Network coding as a performance booster for concurrent multi-path transfer of data in multi-hop wireless networks. IEEE Trans. Mob. Comput. **16**(4), 1047–1058 (2017)
76. Cao, Y., Xu, C., Guan, J., Zhang, H.: CMT-CQA: cross-layer Qos-aware adaptive concurrent multipath data transfer in heterogeneous networks. IEEJ Trans. Electr. Electron. Eng. **10**(1), 75–84 (2015)
77. Adhari, H., Dreibholz, T., Becke, M., Rathgeb, E.P., Tüxen, M.: Evaluation of concurrent multipath transfer over dissimilar paths. In: 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications, pp. 708–714. IEEE (2011)
78. Natarajan, P., Ekiz, N., Amer, P.D., Iyengar, J.R., Stewart, R.: Concurrent multipath transfer using SCTP multihoming: introducing the potentially-failed destination state. In: International Conference on Research in Networking, pp. 727–734. Springer, Berlin, Heidelberg (2008)
79. Fiore, M., Casetti, C., Galante, G.: Concurrent multipath communication for real-time traffic. Comput. Commun. **30**(17), 3307–3320 (2007)

80. Casetti, C., Gaiotto, W.: Westwood SCTP: load balancing over multipaths using bandwidth-aware source scheduling. In: IEEE 60th vehicular technology conference, 2004. VTC2004-Fall. 2004, Vol. 4, pp. 3025–3029. IEEE (2004)
81. Abd, A., Saadawi, T., Lee, M.: Improving throughput and reliability in mobile wireless networks via transport layer bandwidth aggregation. Comput. Netw. **46**(5), 635–649 (2004)
82. Abd, A., Saadawi, T., Lee, M.: LS-SCTP: a bandwidth aggregation technique for stream control transmission protocol. Comput. Commun. **27**(10), 1012–1024 (2004)
83. Argyriou, A., Madisetti, V.: Bandwidth aggregation with SCTP. In: GLOBECOM'03. IEEE Global Telecommunications Conference (IEEE Cat. No. 03CH37489), vol. 7, pp. 3716–3721. IEEE (2003)
84. Wei, W., Xue, K., Han, J., Wei, D.S., Hong, P.: Shared bottleneck-based congestion control and packet scheduling for multipath TCP. IEEE/ACM Trans. Netw. **28**(2), 653–666 (2020)
85. Xue, K., Han, J., Ni, D., Wei, W., Cai, Y., Xu, Q., Hong, P.: DPSAF: forward prediction based dynamic packet scheduling and adjusting with feedback for multipath TCP in Lossy heterogeneous networks. IEEE Trans. Veh. Technol. **67**(2), 1521–1534 (2018)
86. Xue, K., Han, J., Zhang, H., Chen, K., Hong, P.: Migrating unfairness among subflows in MPTCP with network coding for wired-wireless networks. IEEE Trans. Veh. Technol. **66**(1), 798–809 (2016)
87. Khalili, R., Gast, N., Popovic, M., Le Boudec, J.Y.: MPTCP is not pareto-optimal: performance issues and a possible solution. IEEE/ACM Trans. Netw. **21**(5), 1651–1665 (2013)
88. Ha, S., Rhee, I., Xu, L.: CUBIC: a new TCP-friendly high-speed TCP variant. ACM SIGOPS Oper. Syst. Rev. **42**(5), 64–74 (2008). https://doi.org/10.1145/1400097.1400105

Authors and Affiliations

Lal Pratap Verma[1] · Varun Kumar Sharma[2] · Mahesh Kumar[3] · Dimitris Kanellopoulos[4] · Aniket Mahanti[5]

Lal Pratap Verma
er.lpverma1986@gmail.com

Varun Kumar Sharma
varunksharma.102119.cse@gmail.com

Mahesh Kumar
mahesh.chahar@gmail.com

Dimitris Kanellopoulos
d_kan2006@yahoo.gr

[1] Department of Computer and Communication Engineering, Manipal University Jaipur, Jaipur, Rajasthan, India

[2] Department of Computer Science and Engineering, The LNM Institute of Information Technology, Jaipur, Rajasthan, India

[3] Department of Computer Science and Engineering, Jaypee University of Engineering and Technology, Guna, Madhya Pradesh, India

[4] Department of Mathematics, University of Patras, 26500 Patras, Greece

[5] University of Auckland, Auckland, New Zealand