Performant TCP over Wi-Fi Direct

Hanlin Huang[†], Ke Xu[†¶], Xinle Du[†‡], Yiyang Shao[‡], Jie Li[‡], Xiangyu Gao[†], and Tong Li[§]
Tsinghua University[†] Zhongguancun Lab[¶] Huawei[‡] Renmin University of China[§]
Email: {hhl21, dxl18, gxy21}@mails.tsinghua.edu.cn; xuke@tsinghua.edu.cn;
{shaoyiyang, leo.lijie}@huawei.com; tong.li@ruc.edu.cn

2024 IEEE/ACM 32nd International Symposium on Quality of Service (IWQoS) | 979-8-3503-5012-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/IWQoS61813.2024.10682911

***Abstract*—Wi-Fi Direct has been serving a progressively wide range of applications such as device-to-device file sharing, face-to-face interactive gaming, and wireless projection. However, when TCP meets Wi-Fi Direct, we find that two independent control loops exist, i.e., the transport-layer control loop and the link-layer control loop. First, these functionally redundant loops result in spectrum inefficiency. Second, the lack of effective information interaction between layers results in local optimal. To tackle these issues, this paper proposes Wi-Fi Direct TCP (WDTCP), a performant TCP that provides a full protocol design of the *acknowledgment de-redundancy* and *explicit-capacity-based congestion control*. WDTCP tightly couples the two control loops by capturing the WiFi Direct's key feature of one-hop communication. Evaluation results demonstrate that WDTCP can maximize bandwidth utilization while keeping low latency. For instance, compared to legacy TCP, WDTCP improves throughput by up to 49.2% and reduces average and 95th latency by up to 32.4% and 50.7%, respectively.**

***Index Terms*—Wi-Fi Direct, control loop, congestion control**

I. Introduction

Wi-Fi Direct, a wireless technology that enhances direct device-to-device communication, has increasingly been used in recent Internet of Things (IoT) [1, 2] scenarios such as device-to-device file sharing [3], face-to-face interactive gaming [4], and wireless projection [5]. Wi-Fi Direct is based on IEEE 802.11 and allows two wireless local area network (WLAN) devices to establish a direct connection. As a particular case of WLAN, the main difference with the multi-hop network is that its transmission requires only one hop without an intermediate access point (AP) like a router.

The interaction between TCP and multi-hop WLAN is a well-studied topic [6, 7]. Researchers usually optimize TCP for wireless scenarios by modifying the end-to-end congestion control algorithm (CCA) to predict the network condition accurately. However, in such algorithms, two control loops exist, i.e., the transport-layer (L4) control loop and the link-layer (L2) control loop. Firstly, the two control loops suffer from functional redundancy in terms of feedback control. It causes similar media access overhead at the MAC. By sharing the same medium path for ACKs and data packets, frequent ACKs create competition and collisions, wasting wireless resources. Although TACK [8, 9] reduces frequency, the volume of ACKs remains significant under high bandwidth. In addition, the lack of practical information interaction between two loops can only achieve local optimum. For example, CCAs (Cubic [10], Reno [11], Vegas [12]) that use loss and delay as congestion signals are challenging to adapt to the transmission environment of wireless networks. Because these QoS signals are stale information, they cannot explicitly reflect the wireless network transmission state.

Therefore, TCP optimization in current wireless scenarios increasingly tends to adopt the cross-layer ideas [13]. That is, the MAC layer information (channel quality, rate) is passed to the transport layer by modifying intermediate nodes (like AP and base station) to achieve coupling of the transport and MAC (link) layers (e.g., HACK [14], PBE-CC [15], seen in section II) for more accurate control. However, implementing cross-layer optimization for multi-hop WLANs requires modifying intermediate nodes, which are usually uncontrollable for developers [16].

Fortunately, the one-hop feature of Wi-Fi Direct naturally supports cross-layer optimization [17]. But existing multi-hop WLAN cross-layer schemes are still far from satisfactory for Wi-Fi Direct. For example, HACK [14] suffers from feedback lag, which might result in inaccurate control. While PBE-CC [15] is specially customized for cellular network transmission, it cannot be used for Wi-Fi Direct. Moreover, TCP transfer usually requires combining two components: an acknowledgment mechanism and a congestion control. The acknowledgment mechanism ensures the integrity of data transmission by retransmitting the lost packets. Congestion control is to adjust the transmission rate when the link condition changes, ensuring high link utilization and low latency. From this perspective, HACK mainly focuses on the acknowledgment mechanism while PBE-CC mainly focuses on congestion control.

For a performant TCP over Wi-Fi Direct, this paper proposes WDTCP. WDTCP tightly couples the L4 and L2 control loops and provides a full protocol design that considers both acknowledgment mechanisms and congestion control. Specifically, WDTCP comprises two main components: Acknowledgment De-redundancy (AD) and Explicit-capacity-based Congestion Control (ECC). The former removes ACKs from the transport layer, reducing competition with data frames for wireless resources. The latter uses channel capacity, the most intuitive congestion signal, which enables accurate control.

AD takes advantage of Wi-Fi Direct's one-hop transmission

This work is supported in part by the National Key Research and Development Program of China (No.2022YFB3102301), the NSFC Projects (No.61932016, No.62132011, No.62221003 and No.62202473), the China National Funds for Distinguished Young Scientists (No.62425201). Corresponding author: Tong Li and Ke Xu.

and eliminates the need for transport layer acknowledgment by reusing the BLOCK-ACK (BA) mechanism of the Wi-Fi MAC protocol. By obtaining lost frames at the MAC layer, the transport layer directly retransmits the corresponding packets without requiring additional acknowledgments from this layer. It reduces medium contention and protocol overhead, providing high channel utilization.

ECC is a rate-based control strategy instead of a window-based one, combining a Multiplicative Increase Multiplicative Decrease (MIMD) scheme and an Additive Increase Additive Decrease (AIAD) scheme. It can adapt to dynamic changes in the wireless channel by switching between MIMD and AIAD. Specifically, the sending rate is multiply adjusted according to the channel capacity measured at the MAC layer to ensure faster convergence speed. Meanwhile, to control the transmission delay, the MAC queue size is monitored and the rate is linearly adjusted, which ensures low-latency transmission in the premise of high bandwidth utilization.

Our evaluation shows that WDTCP provides higher throughput and lower latency over Wi-Fi Direct compared to legacy TCP. In particular, the average throughput of WDTCP is 49.2% at 160MHz and 36.7% at 80MHz, higher than TCP's. These gains in throughput are mainly attributed to AD, which removes redundant acknowledgment from the L4 layer and reduces competition for wireless resources. Therefore, WDTCP makes better utilization of the channel bandwidth. Additionally, the transmission latency is reduced by 32.4% on average and is smoother without substantial jitter. It is also demonstrated that the applications adopting WDTCP converge faster, even under fluctuating channel changes. This is because ECC adopts a mixture of the MIMD and AIAD schemes using physical link capacity, the intuitive congestion control signal, as the basis for rate adaption.

The rest of this paper is organized as follows. In the next section, we talk about related works. Section III describes current interaction problems between TCP and Wi-Fi Direct and the importance of optimizing it. Section IV introduces our proposed solution WDTCP, which optimizes the transmission process with the characteristics of Wi-Fi Direct. Section V depicts the key components and how to set corresponding parameters. Following that, we implement our approach and perform experiments on throughput, latency and fairness of WDTCP. Finally, we conclude the paper in section VII.

II. RELATED WORK

In this section, we first overview single-layer and cross-layer TCP optimizations over multi-hop wireless links. Then we discuss the research status of TCP optimization over Wi-Fi Direct.

Single-layer TCP optimization. Existing state-of-art [6, 7, 18, 19] optimize TCP mainly with the help of loss, delay, and other metrics. They sense the wireless network state through these QoS signals and adjust the sending rate accordingly. Verus [18] is an end-to-end congestion control protocol for wireless cellular networks. It uses delay measurement to respond quickly to capacity changes without predicting the channel dynamics in the cellular network. Sprout [19] uses the packet arrival time measured at the receiver as a congestion signal to determine network congestion, and the sender uses probabilistic reasoning to make short-term predictions of the bottleneck link rate. These strategies have demonstrated a degree of effectiveness in optimizing LTE, and their application can also be extended to multi-hop WLAN scenarios. However, it is important to acknowledge that these signals suffer from inherent inaccuracies, resulting in either untimely rate adjustments or slow convergence.

Cross-layer TCP optimization. To avoid using inaccurate end-to-end signals, many cross-layer-based optimization proposals [20, 21, 14, 15] have been proposed. For example, HACK [14] eliminates resource contention by carrying ACKs at the link layer. Whenever the link layer at the receiver generates a BA, it writes information about ACKs from the previous round of TCP into the current BA frame. It avoids separate transmission of ACKs and significantly improves media utilization. Nevertheless, the trigger time of link layer ACKs and transport layer ACKs are asynchronous, which is likely to cause acknowledgment delay. PBE-CC [15] uses the channel bandwidth as the basis for speed regulation. By measuring the bandwidth of the link and tracking changes in available capacity of the channel, it enables more precise control over the sender's rate. But it can only serve cellular networks that use resource block allocation [22]. It is worth noting that WDTCP differs from both HACK and PBE-CC. While the former solely focuses on acknowledgment mechanisms, the latter exclusively addresses congestion control considerations.

TCP optimization over Wi-Fi Direct. Compared with the studies on Wi-Fi Direct that focus on application services (e.g., file sharing, interactive gaming, wireless projection [3, 4, 5]), neighbor discovery [23], or ad-hoc networking in group communication [24], much fewer studies explore how the transmission protocol (e.g., TCP) interacts with WiFi Direct. Among them, Wang [24] presents an intra-group communication system and constructs an inter-group bi-directional communication solution using Wi-Fi Direct's characteristics. Sun proposes LCR [23], a simple and effective way to expedite the device discovery procedure. None of them address the competition for resources due to two control loops. Fortunately, Wi-Fi Direct's one-hop feature is a natural fit for cross-layer optimization of TCP. Without the help of intermediate nodes, WDTCP can reduce the competition of acknowledgment mechanisms and enable accurate congestion control, which provides a complete design opportunity.

III. BACKGROUND AND MOTIVATION

A. Transmission Performance is Far from Satisfactory over Wi-Fi Direct

Wi-Fi device-to-device communication, also known as Wi-Fi Direct, is a technology launched by Wi-Fi Alliance [25] that allows direct connection between two devices based on the original Wi-Fi technology. It allows users to communicate one-to-one or one-to-many without the aid of LAN or AP. Wi-Fi Direct has been serving a progressively wide range of appli-

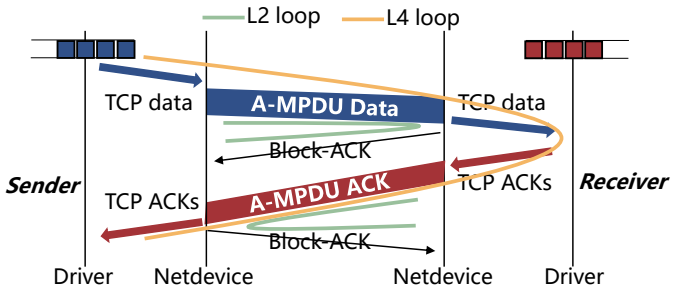


Fig. 1: Two control loops in TCP over WLAN.

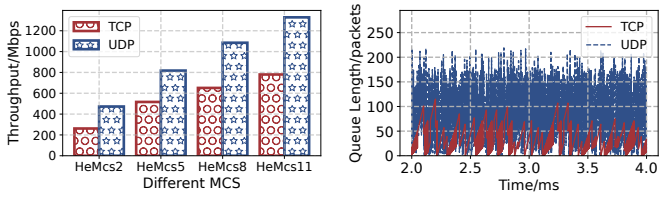
cations like throughput-intensive device-to-device file sharing (e.g., Huawei Share [26], Airdrop [27]) or delay-sensitive face-to-face interactive gaming (e.g., P2P Pong [4]). Recent years have also witnessed the rapid growth of video-based applications over Wi-Fi Direct like wireless projection [5], which demands both high throughput and low latency.

However, the bandwidth provided by TCP for Wi-Fi Direct is challenging to reach the upper bound of WiFi. It is reported that [8, 14] with 64-QAM modulation mode and 40MHz channel width, the physical capacity of IEEE 802.11n can reach 300Mbps theoretically. The UDP applications based on it can reach 210Mbps, while TCP can only provide an ideal throughput of 160Mbps, half of the theoretical value. Furthermore, there is room for optimization in TCP's latency control. According to the experimental Figure 8(b), the tail latency also performs poorly. The one-way delay reaches a maximum of 2.5ms for the 95th latency, and Figure 8(c) also demonstrates that the delay jitter is more evident under a sending rate of 300 Mbps.

B. TCP Stack is Redundant over Wi-Fi Direct

TCP transmission over Wi-Fi Direct needs to go through two control loops: the L4 control loop and the L2 control loop. Take Figure 1 as an example. In the L4 loop, the sender generates data packets at the transport layer, which are sent to the channel by the NIC in the form of A-MPDU aggregation. The receiver NIC receives the aggregated frame and passes it to the upper layer. Then the transport layer generates ACKs accordingly and hands them to the NIC, which are also sent to the sender in the form of A-MPDU. For the L2 loop, it needs to generate BA for both data and ACKs, respectively. Two L2 loops are required in a single L4 loop. However, during the L4 loop when the receiver sends the ACKs, the channel is occupied. As a result, the sender can only send the next A-MPDU data frame after receiving the TCP ACKs. Since the transmission opportunity cannot be obtained in time, the data frame carrying payload will back off for the next sending opportunity. In a complete transmission process, many ACKs will be generated, even accounting for half of the total number of packets. This increases protocol overload and leads to longer task completion time.

We demonstrate the redundancy through a set of pre-experiments. We construct a pair of Wi-Fi Direct nodes where the sender contains multiple services that initiate requests according to a Poisson process. TCP Cubic and UDP are tested



(a) Throughput under MCS. (b) Queue length evolution.

Fig. 2: The redundancy in two control loops

at the nodes, respectively. Figure 2(a) shows the results under different Modulation Coding Scheme (MCS) [28], where MCS affects the link capacity. It is easy to see that UDP outperforms TCP in terms of throughput. Especially under HeMcs11, UDP is up to 500Mbps higher than TCP. Therefore, without two control loops, UDP can utilize the link capacity more fully. However, adopting UDP directly is undesirable. Besides its unreliability, it lacks rate control and packets are accumulated in the Wi-Fi MAC queue (shown in Figure 2(b)), leading to higher latency.

C. End-to-end Congestion Signals are Inaccurate

End-to-end congestion channels do not explicitly reflect the wireless transmission condition, which makes TCP face a bottleneck in providing high throughput and low latency in wireless environments. These signals are QoS data metrics that do not consider wireless interference. Yet wireless interference is an outstanding feature of Wi-Fi. This can lead to unexplicit control of CCAs based on end-to-end congestion signals, with the following main reasons.

High bit error rate: The wireless network link is a lossy medium. Due to channel interference, multipath fading and other reasons, wireless network links have a high bit error rate compared with traditional wired networks. It will result in packet damage or loss, causing the sender to initiate congestion control. From the report [29], if the loss occurs before congestion, loss-based CCAs result in low throughput [30, 31, 32].

Unpredictability of delay: Because mobile users move randomly, the distance from the base station differs, bringing about different time delays. The unpredictable delay also affects the accuracy of TCP congestion determination [33].

Fluctuated wireless capacity: Nodes can join and leave the WLAN at any time, which can bring an increase or decrease in the available link capacity. The TCP sender is slow to perceive this situation, and the throughput cannot converge in time [34].

Above are the three main factors that affect TCP CCAs' judgment, causing low link utilization and slow convergence.

D. Time to Revisit TCP Optimization over Wi-Fi Direct

Unlike WLAN and cellular networks that require multi-hop propagation, Wi-Fi Direct works only for one-hop transmission, which provides new opportunities for a minimalist protocol design. In this paper, we revisit TCP optimization over Wi-Fi Direct in a cross-layer paradigm.

First, instead of providing duplicate acknowledgments in both the L4 control loop and the L2 control loop, we can remove one of them by carefully transferring necessary states

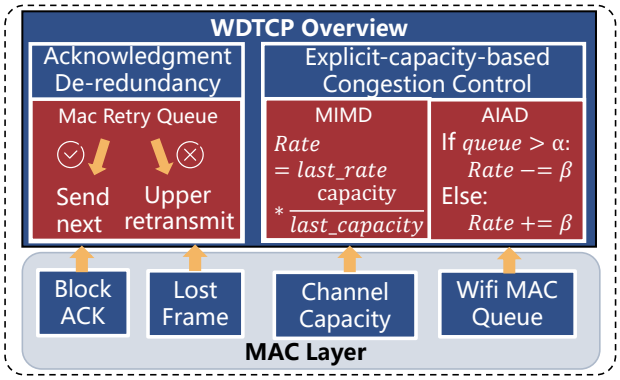


Fig. 3: The architecture of WDTCP

between L4 and L2. For example, in this paper, we remove all TCP ACK at the L4 and only adopt the L2 acknowledgments. With the feedback from L2, L4 is only responsible for retransmitting lost packets without sending any TCP ACKs. This significantly reduces wireless medium access overhead from L4 ACKs.

Second, instead of using the L4 end-to-end congestion signals (e.g., delay, loss), we can enable the interaction between L4 and L2. For example, in this paper, we use the physical link capacity measured at L2, the most intuitive indicator of the channel condition, as the basis for L4's pacing regulation. Based on this L2 indicator, a specific congestion control can therefore be carefully designed.

IV. Overview

As discussed earlier, current TCP-based transmission over Wi-Fi Direct has two control loops, which suffer from redundant acknowledgment feedback and a lack of practical information interaction. To address the problem, We optimize the transmission control protocol from two perspectives. On the one hand, we remove the need for transport layer's ACKs by multiplexing the BAs at the MAC layer. More transferring opportunities are given to useful payload to improve transmission throughput. On the other hand, to adapt to fluctuations in wireless bandwidth, we reconstruct congestion control. Instead of using signals based on QoS metrics, the current wireless capacity is measured at the end host to reflect the real-time channel condition directly. Combining both, we propose a new optimized design for Wi-Fi Direct—WDTCP.

Figure 3 shows the main components of WDTCP: Acknowledgment De-redundancy (AD) and Explicit-capacity-based Congestion Control (ECC). The former removes the acknowledgment mechanism of the transport layer. By reusing BA information of the MAC layer, we can determine which frames are successfully received and which are lost. ECC will notify the transport layer to retransmit corresponding packets only when the retransmission of the MAC layer still fails. The module keeps measuring the wireless capacity and reports it to the transport layer in time. When the changes in capacity are sensed, the application sending rate is adjusted dynamically.

WDTCP is a cross-layer architecture protocol. We use the proprietary protocol at the transport layer to reduce channel

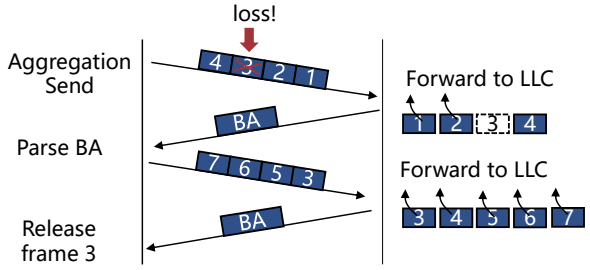


Fig. 4: Frame aggregation and acknowledgment

contention as much as possible. Information from the MAC layer is used to obtain the physical capacity, queue length, BA, etc., which are notified to the transport layer for packet retransmission and congestion control. The AD module takes advantage of one-hop and low loss rate of Wi-Fi P2P scenarios. By eliminating redundant transport layer acknowledgments and obtaining information about lost packets through the MAC layer, the medium competition is significantly reduced while ensuring reliable transmission. In ECC, multiplication adjustment can quickly adapt to the changes in the physical channel. Linear adjustment can provide low delay and make full use of the wireless channel to avoid under-utilization. It should be noted that WDTCP can only be used in single-hop transmission scenarios like Wi-Fi Direct. In the multi-hop network, the acknowledgements of transport layer's packets cannot be delivered to the sender because the MAC layer does not support IP routing.

V. Design of WDTCP

A. Acknowledgment De-redundancy

Since WDTCP multiplexes the BA mechanism of the MAC protocol, we first briefly introduce the acknowledgment process of data frames at the MAC layer. On this basis, we carefully analyze how to apply this process to determine the packets that need to be retransmitted.

1) Aggregation Acknowledge: In the complete frame transmission process, the transmission efficiency can be improved by frame aggregation. Accordingly, the traditional regular ACK can be replaced by BA introduced in 802.11e, which replies to acknowledgments of multiple frames simultaneously, thus improving resource utilization.

We use an interactive example to illustrate the aggregation process in Figure 4. The A-MPDU sent by the client contains 4 data frames. Assuming that subframe 3 is lost, the server responds BA for subframes 1, 2 and 4. It buffers subframe 4 temporarily, waiting for the frame with a smaller sequence number. After the client parses the BA, it sends a new aggregate frame containing subframe 3, meanwhile maintaining a counter for it. When the server receives the new frame, it reorders these five frames and forwards them to Logic Link Control (LLC) layer. Then the client receives ACK for subframe 3 and releases it. However, if the number of retransmissions reaches the limit or a timeout occurs, subframe 3 will be discarded even if it is not accepted successfully.

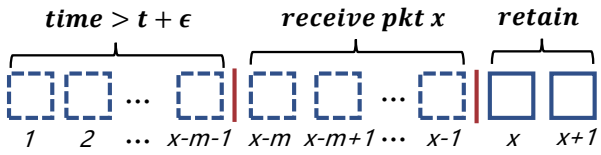


Fig. 5: Two cases for dropping packets

2) Multiplex Block-ACK: WDTCP achieves high throughput by eliminating the transport layer acknowledgment mechanism and utilizing the MAC layer's BA. Upon receiving the BA, the client can determine which frames have been successfully accepted and which ones require retransmission. Frames in need of retransmission are placed in a retry queue, and a counter and a timer are initiated for each frame. If the corresponding ACK is successfully received after the retransmission, the retransmitted frame is considered successful. Otherwise, it is sent again until the retry count reaches the limit or times out. As for the packet that needs to be retransmitted, the failed frame can be detected locally and the sequence number or ID of the packet corresponding to the frame is notified to the transport layer. The transport layer retains the packets to be acknowledged until it receives the retransmitting signal or times out. We still take Figure 4 as an example to illustrate this process:

Assuming that the channel condition is poor for some time, both subframes 3 and 4 are lost, but the retransmission of subframe 3 is successful, while 4 fails. Then the client's MAC layer continues to retransmit subframe 4, and the ACK of it is still not received after a period of time. At this time, the maximum number of retransmissions has been reached, and the client decides to drop it. At this point, we need to send the packet sequence number corresponding to subframe 4 to the transport layer to inform it of retransmitting. We suppose that the sequence number of the corresponding packet is x. When the transport layer receives the sequence number x, it knows that the previous packets have been successfully accepted, so it can drop $x-1, x-2, ..., x-m$ packets, as shown in Figure 5. Then the packet x is retransmitted, the MAC layer receives it and sends out a new frame containing it. Now the channel is good and the frame is received successfully.

However, it is essential to note that if there is no frame loss, does the transport layer need to cache the sent packets all the time? Keeping all packets is not practical for memory [35]. Therefore we can set a $t+\epsilon$ based on the MAC layer timeout time t. When this time is exceeded, the transport layer can drop the packet. Figure 5 also shows that except for packets $x-m \sim x-1$ dropped due to receipt of retransmission sequence number x, packets $1 \sim x-m-1$ have been dropped due to timeout before. This ensures that packets to be retransmitted are still cached even after MAC layer timeouts, while packets transmitted successfully at the MAC layer can wait for $t+\epsilon$ and then be dropped without concern. The whole process above is simply shown in the first procedure of Algorithm 1.

We now discuss how to obtain the above parameters. The number of retransmissions for a given MSDU is limited, and the frame would be dropped if it reaches the limit [36]. IEEE 802.11 standard proposes the value 7 for the retry number. In the other case, it will also be discarded when the frame stays in the retransmission queue for more than a certain threshold $dotLifetime$ [37]. 802.11n specifies 500ms for it, which can also be obtained from the device. Therefore, the transport layer needs to set the time to keep each packet as $dotLifetime + \epsilon$, ensuring that after the MAC layer loses the frame due to timeout, the transmission layer still retains the corresponding packet for retransmission.

Algorithm 1 WDTCP protocol process

FA: the lost data frame.
S: the packet set buffered in transport layer.
BA: most recently received BLOCK-ACK.

procedure RETRY(FA) ▷ Retransmit the lost frame
  MAC Layer retransmit first
  if Retry still fail **then**
    Parse packet number x corresponding FA
    Upper Layer retransmit packet x
  Update buffer packet Set S
  Discard successfully accepted packets
procedure F$_{rate}$(BA) ▷ Calculate new rate
  Parse BA frame
  Measure rtt
  Calculate current capacity C_{curr}
  $R_{curr} \leftarrow \frac{C_{curr}}{C_{last}} * R_{last}$
  if $q_{len} < a$ **then**
    $\beta \leftarrow \frac{(a-q_{len})}{rtt}$
    $R_{curr} \leftarrow R_{curr} + \beta$
  else if $q_{len} > b$ **then**
    $\beta \leftarrow \frac{(q_{len}-b)}{rtt}$
    $R_{curr} \leftarrow R_{curr} - \beta$
  Update a, b according to C_{curr} and rtt
  $R_{last} \leftarrow R_{curr}$
  $C_{last} \leftarrow C_{curr}$

B. Explicit-capacity-based Congestion Control

In this subsection, we introduce rate-based congestion control. By modeling the data transmission process, we illustrate the relationship between the sending rate and the channel capacity and how to calculate the capacity. ECC dynamically adjusts the sending rate by multiplicative and linear means.

1) Capacity Model: There is a gap between the transmission rate of the physical layer and the actual application rate [38, 39]. After data packets reach the MAC layer, they will not be delivered to the channel immediately. Instead, they have to go through the process as shown in Figure 6, including MPDU encapsulation, DIFS and backoff waiting. Nevertheless, only the MSDU containing packets reflects actual application throughput, and the rest can be treated as MAC's overhead.

We can consider approximately that $throughput \approx \frac{t_{msdu}}{t_{msdu}+t_{overhead}} * capacity$. If an aggregation mechanism like A-MPDU is adopted, the efficiency would be improved, attributing that more MSDUs can be included in a single frame.

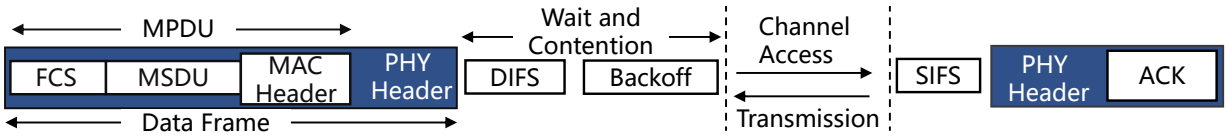


Fig. 6: Data frame and ACK transmission process

Many factors, especially modulation mode, influence wireless capacity. We take the most widely used one, OFDM, as a representative to illustrate wireless capacity. It calculates the capacity (C) according to the following formula [40].

$$C = (N_{subcarrier} * Bit * \mu)/T_{symbol} * N_{stream} \quad (1)$$

where $N_{subcarrier}$ refers to the number of subcarriers. OFDM uses orthogonal subcarriers, decomposing the whole wireless channel into many subcarriers. Each subcarrier can carry Bit bits data for one transmission, and μ determines the coding bitrate to avoid frame fault. We can simply understand the bitrate as transmission redundancy. For example, when $\mu = 1/2$, it transmits one-bit data twice to provide backup for the corrupted packets. Bit and μ both depend on MCS, which would vary with the channel environment. Actually, a physical frame consists of many symbols. MCS determines how much payload a symbol can carry, and each subcarrier sends symbols every T_{symbol} microseconds. The symbol period is fixed for each Wi-Fi standard. In addition, antennas are introduced in Multiple-Input Multiple-Output (MIMO) to enhance performance. The number N_{stream} of antennas affects the physical rate proportionally. Together, all the above factors affect the wireless capacity complying with Equation (1).

2) Adaptive Adjustment: WDTCP senses the state of the physical channel bandwidth based on the measured wireless capacity. The sender can detect Bit and μ by decoding the SIGNAL field of Preamble header [41], and then calculate the current capacity (C_{curr}) according to Equation (1). After passing C_{curr} to the transport layer, it is compared with the last obtained capacity C_{last}. The ratio α of the change is calculated ($\alpha = \frac{C_{curr}}{C_{last}}$) and the newest rate $R_{new} = \alpha * R_{last}$, R_{last} is the last sending rate. Thus the application rate is adjusted in proportion to the change in the physical capacity.

However, the multiplicity adjustment is coarse-grained after all. For higher channel utilization and lower latency, we also fine-tune it through AIAD. Specifically, data packets are queued at the MAC layer before being sent to the wireless channel, later sent in the form of aggregation. Therefore, the MAC queue starts to build up. We must keep the queue length in a reasonable range. Too low a queue length will not fully utilize the channel, hence throughput is affected. While too high a queue length will result in higher latency. This is because the number of data packets aggregated at one time is limited. If there are too many data packets in the queue, some packets will not be sent out in the current round. They will wait longer in the queue, leading to increased latency.

We tentatively assume that the reasonable interval of the queue length is $[a, b]$. When the real-time queue length of less than a is detected, it will linearly increase from the original rate. When the length is larger than b, it will decrease linearly. The transmission delay and queue length determine the degree of linear change, which will be illustrated in the following subsection. As for regulating the transmission rate, we can control the number of bytes sent per unit of time or the sending interval of each packet. The pseudo-code of control algorithm is shown in Algorithm 1 (the second procedure).

From the above process, if the channel conditions are relatively stable, the pacing rate will not change significantly, and the MAC queue length will remain in a reasonable range. Once the channel condition changes, WDTCP senses it quickly and adjusts the rate to a fair share of the bandwidth.

3) Parameter Analysis: We now establish a multi-pair device coexistence model to analyze the state transition of WDTCP. Considering that there are n pairs of devices sharing Wi-Fi channels, according to the OFDM frequency division multiplexing mechanism [42], only one pair of nodes can communicate at each time. The channel access is probed before transmitting. If the channel is busy at this time, it will back off following CSMA/CA rule. As mentioned earlier in section III, the backoff time caused by unsuccessful attempts is multiplied by 2. Thus the chance of successfully gaining access to the channel is inversely proportional to the number of competing nodes. Suppose that the nth pair of nodes stops using the channel at this point, 1/n of the channel accesses are released and distributed equally to the remaining n-1 pairs of nodes. Without losing generality, we think the average backoff time of the first pair decreases from bf_1 to bf_2, where bf_1 and bf_2 satisfy the following relationship.

$$bf_2 = \frac{n-1}{n} * bf_1 \quad (2)$$

To make full use of the free bandwidth left by the node withdrawal, the Wi-Fi MAC queue must contain at least q_{min} bytes at this time.

$$q_{min} = a = \left(\frac{bf_1}{bf_2} - 1\right) * BDP \quad (3)$$

$$BDP = C_{curr} * rtt \quad (4)$$

where BDP is channel capacity-delay product. Too many packets stored in the queue will cause accumulation. According to the aggregation principle, at most Agg_{max} bytes of data can be aggregated at one time. Agg_{max} is approximately 8K in A-MSDU, 64K in A-MPDU. Therefore, the maximum queue length q_{max} (b) is set to Agg_{max} to prevent excessive data packets from not being sent and incurring a long dwell time.

AIAD is a linear fine-tuning to allow the application rate to match the physical bandwidth fully. Based on the delay of data frames in one transmission, we have

$$rtt = t_{DIFS} + t_{backoff} + t_{SIFS} + t_{data} + t_{ack} \quad (5)$$

$$\beta = \frac{\delta_{qlen}}{rtt} \tag{6}$$

$$\delta_{qlen} = \begin{cases} a - q_{len} & q_{len} < a \\ 0 & a \le q_{len} \le b \\ q_{len} - b & b < q_{len} \end{cases} \tag{7}$$

We can refer to Figure 6 to know the meaning of each notation about time, except for t_{data} and t_{ack} are the time of data frame and ACK transmission respectively. When the queue length (q_{len}) is not in that range, we need to adjust the rate linearly at step of β to bridge the gap with the normal queue length quickly. Specially, if q_{len} is less than a, we increase sending rate by β. Otherwise, the rate is decreased by β according to (6) and (7).

As for the initial sending rate, we cancel the slow start and use a higher rate as the initial speed. Even if the initial rate does not match $\eta * C$, we can detect this state by the queue length and then adjust the rate to a suitable range in a short time. By probing the queue, WDTCP can utilize the channel bandwidth more quickly. The results in section VI show that WDTCP can make full use of the remaining bandwidth immediately and keep the delay at a low value.

VI. Evaluation

First of all, we note that the current work is still in the verification stage, primarily due to the absence of programmable chips on the mobile side that meet the performance requirements. We firmly believe that further experimentation is necessary for comprehensive verification, including deployment on mobile NICs. However, we consider this aspect as part of our future work and leave it for further investigation. In this paper, we implemented WDTCP on the ns-3 platform instead. The MAC layer obeys IEEE 802.11 standard and the transport layer adds rate control logic based on raw socket.

The MAC layer is the main region to obtain effective information. When the corresponding BA is received, the packet is parsed to obtain the SIGNAL field (mentioned in section V). Then the current physical capacity C is calculated following Equation (1). Subtracting the time when the frame is sent from the current receiving time can obtain the frame round-trip delay rtt. A trigger function is set to record the current backoff time whenever the competition for the wireless channel is successful. In addition, there are two monitoring processes that monitor the MAC queue length and the packet's status in the retransmission queue. When the queue length is outside the range, calculate β by Equation (6) and immediately notify the transport layer of linear speed regulation.

We focus on evaluating the performance of WDTCP and comparing it with the legacy TCP. Due to space limitations, in order to show each assessment metric in detail, we only use Cubic [10] algorithm in the previous experiments. In the last subsection, we compare with other algorithms, such as BBR [43], NewReno [11], and Vegas [12]. The Wi-Fi standard used in our experiments is the latest 802.11ax, with two antennas supported. The other settings will be tailored to the specific experiment. Four critical aspects are evaluated:

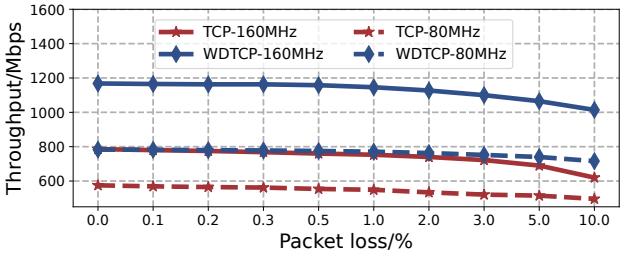


Fig. 7: Throughput under different packet loss.

Key performance. We firstly test the most important performance metrics, including throughput and latency under different loss and load.

In-depth understanding. WDTCP's adaptability to Wi-Fi changes relies on good convergence, and the complete design comprising AD and ECC ensures high performance.

Bandwidth fairness. Although WDTCP adjusts speed in a more aggressive way to make full use of channel bandwidth, it does not generate malicious competition.

Comparison with other schemes. We compare it with UDP and other TCP CCAs respectively to extensively prove the advantages of WDTCP in different scenarios.

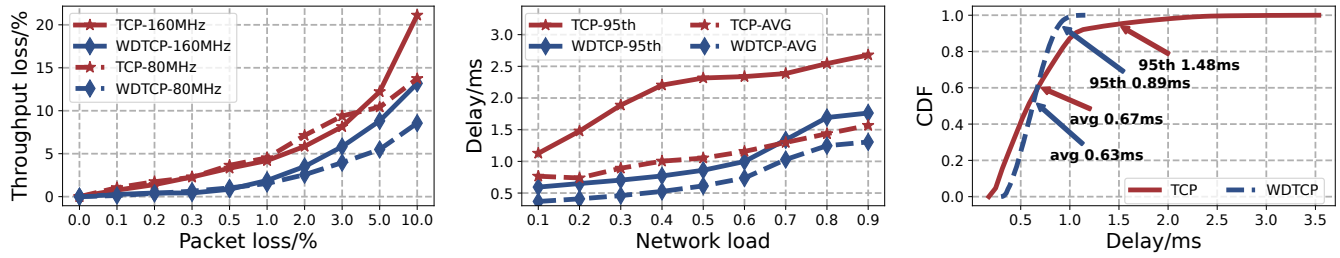
A. Key Performance

The first step is to verify that the customized protocol WDTCP can achieve high throughput and low latency, which are the most critical metrics for application requirements (see section III). According to the literature [44], it is noted that Peer-to-Peer (P2P) scenarios have at most 5% packet loss at close range. Therefore, apart from regular lossless communication, we also explore the throughput variation when the packet loss rate is from 0 to 10%.

For the throughput test, We have carried out experiments on the frequency width of 80MHz and 160MHz under one pair nodes respectively and explored the change of throughput in the case of packet loss. The results are shown in Figure 7. It can be seen that under 160MHz, the maximum throughput with WDTCP protocol can reach 1160Mbps, while TCP's is only 790Mbps, a 49.2% improvement of WDTCP over TCP. Under 80MHz, WDTCP improves 36.7% over TCP. This is mainly because WDTCP removes redundant transport layer ACKs, which reduces competition for wireless resources and leaves more space for payload transmission.

In the presence of packet loss, when the packet loss rate is less than 1%, the throughput of both does not drop significantly, mainly benefiting from the retransmission of the MAC layer. However, when the packet loss rate is 1%∼10%, both throughputs have a different degree of decline. This is because the transmission layer of WDTCP and TCP both need to retransmit the lost packets, resulting in some overhead. Nevertheless, WDTCP can still maintain a high throughput of 1020 Mbps when the packet loss rate is 10%.

Based on the above results, we compare the variation of the throughput loss rate with packet loss for WDTCP and TCP, respectively. Namely, each throughput is compared only with the relative value of its protocol throughput to guarantee



(a) Throughput loss rate under different packet loss. (b) Avg and 95th one-way delay comparison. (c) One-way delay under 300 Mbps.

Fig. 8: Throughput loss and delay comparison between TCP and WDTCP

fairness. From Figure 8(a), the throughput loss rate starts to increase when the packet loss rate exceeds 1%. Still, the loss rate of WDTCP is lower than TCP's, and the improvement is more pronounced as the packet loss rate increases. The higher loss rate of TCP is because its retransmission is based on the ACK signal from the transport layer at the receiver, which has a high overhead. Compared with WDTCP, the reaction is slower. This experiment proves that WDTCP is tolerant of channel packet loss, and the protocol is robust.

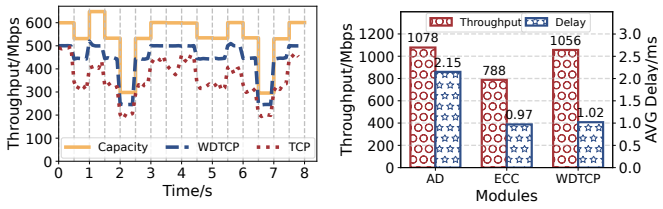
Next, we add 8 senders and make requests to the same receiver with different network loads. We evaluate the packet one-way delay of TCP and WDTCP. We have measured the average and 95 percentile delay for both under different application sending rates. As seen in Figure 8(b), WDTCP is lower than TCP for each type of latency comparison. Specifically, WDTCP reduces the average latency compared to TCP by 32.4% and 95th latency by 50.7% on average. The reason why WDTCP can provide lower latency is that it avoids packet accumulation by controlling the MAC queue length. In contrast, TCP's congestion control generates stacking at the transport and MAC layers, resulting in higher latency.

To explore the jitter, we test the packet delay-timing sequence when the sending rate is 300Mbps. Figure 8(c) shows that the latency of WDTCP fluctuates within 1ms. Nevertheless, TCP has many abrupt changes, even up to 3.5ms, which is unfriendly, especially for applications with low tail latency requirements. It proves that WDTCP is smoother, while TCP has higher delay fluctuations due to queuing and bursting.

B. Deep dive into WDTCP

1) Convergence speed: Through rate-based adaptation, the application sending rate of WDTCP can match the physical channel bandwidth. To simulate the drastic changes in the channel, we change MCS every 0.5s and run the experiment for 8s. Dynamically changing MCS is also the way many applications nowadays adapt to the channel condition [45]. We still conduct this test based on the above device-to-device scenario with a frequency width of 160MHz.

Figure 9(a) shows the application throughput of WDTCP and TCP in the case of drastic changes in channel capacity, where the channel capacity (remarked by yellow) is the maximum theoretical value described in section V. It is easy to see that WDTCP can converge faster and maintain a higher throughput. TCP, on the other hand, converges slowly and has more significant fluctuations in application throughput. In



(a) Throughput varies with capacity. (b) WDTCP's full design.

Fig. 9: (a) When the channel changes drastically, WDTCP can converge quickly. (b) AD and ECC work together to obtain high throughput and low latency.

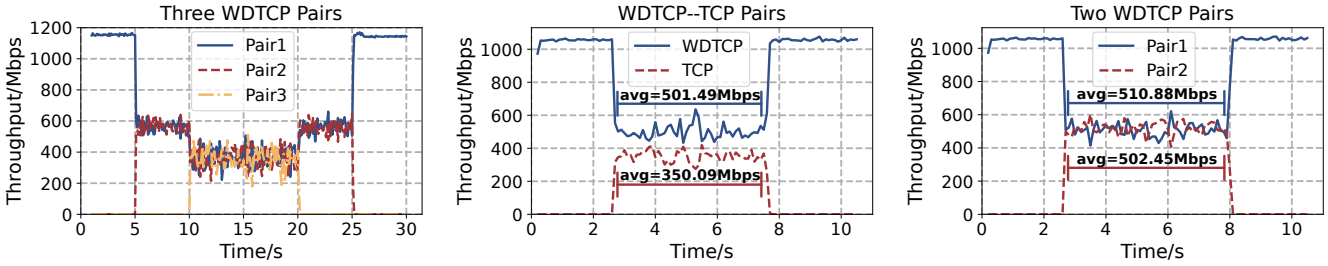
0∼8s, we take every 0.5 seconds as a time interval. It is not difficult to see that the throughput of WDTCP within each interval is also more stable than that of TCP. The reason behind these gains is that WDTCP is directly tuned to the physical channel capacity, which is more timely than indirect metrics such as RTT used by Cubic. Moreover, the linear adjustment also makes WDTCP's throughput more stable.

2) Complete design: As described in section III, the collaboration of acknowledgment mechanism and congestion control is necessary for a complete protocol design. To demonstrate the need for both AD and ECC, we perform ablation experiments for each module. Figure 9(b) illustrates the results. With the presence of AD only, we eliminate redundant loops, which greatly improve the throughput. However, the average latency is higher due to the lack of rate control. Conversely, with ECC only present, it guarantees a shorter queue. But repeated acknowledgments increase the protocol stack overhead. Combining AD and ECC, WDTCP achieves high throughput and low latency. It demonstrates the importance of cooperation between two modules.

C. Bandwidth Fairness

We conduct the following experiments to demonstrate that WDTCP's rate adjustment does not negatively affect fairness. Three pairs of communication devices with similar propagation delays are tested. Each pair of communication devices joins at different times, sharing the channel bandwidth.

Figure 10(a) shows that when the second pair of nodes join, they and the first pair can quickly achieve equal throughput. When the second pair ends communicating, the first can also ramp up free bandwidth quickly. The joining and leaving of the third pair also present a similar result. Jain's fairness index [46] is 99.72% and 99.41% with two and three concurrent



(a) Three WDTCP flows. (b) One WDTCP flow coexists with one TCP flow. (c) Two WDTCP flows.

Fig. 10: Fairness between TCP and WDTCP.

flows (100% is ideal), respectively, proving that WDTCP can share bandwidth fairly without affecting other nodes. Because CSMA/CA ensures that each pair has equal access to the channel, WDTCP can quickly adjust the rate fairly and ramp up when there is free bandwidth.

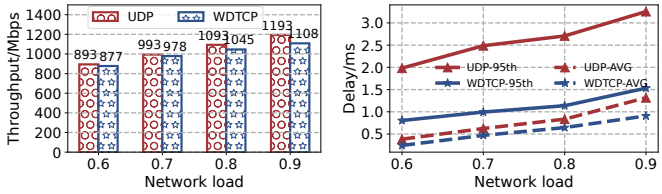
A common requirement for new congestion control schemes is sharing the available bandwidth fairly with existing CCAs. So we explore fairness when coexisting with devices using TCP. Figure 10(b) shows that the throughput is 1030 Mbps when only one pair of WDTCP nodes exists. After a pair of TCP nodes join at 2.5s, WDTCP can reach a fair share of about 501 Mbps, while TCP can only reach about 350 Mbps. Under the same conditions, we also explore a case of two pairs of WDTCP nodes sharing the channel. In Figure 10(c), both pairs can achieve equal throughput when the second pair joins, and their throughput is about 500 Mbps. From figures 10(b) and 10(c), WDTCP can make full use of the bandwidth of the fair share, while TCP's utilization for the wireless channel is low. The main reason why TCP does not achieve higher throughput is the redundancy of its protocol stack, not that WDTCP takes up more channel access opportunities.

D. Comprehensive Assessment

To fully prove good performance WDTCP brings, we also compare it with other schemes in complex scenarios.

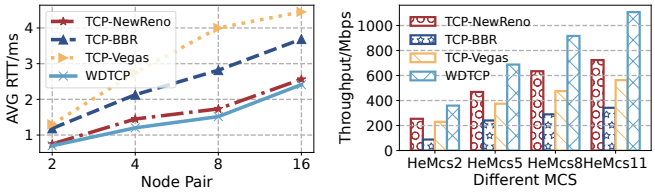
1) WDTCP vs. UDP: Intuitively, using UDP will naturally save double-loop overhead, but it tends to cause packets to queue at the source when the link load is high. Figure 11 shows the results of the comparison with UDP, where we examine the performance of both under high load. In terms of throughput, our scheme has a slight disadvantage, e.g., WDTCP's throughput is just 92 Mbps lower than UDP's when the load is 0.9. This is due to the fact that our retransmission and speed control consume some transmission resources. But the improvement of WDTCP on latency is significant. WDTCP improves up to 42.3% (1532us for WDTCP, 2655us for UDP at 0.9 load) for the 95th delay. For average delay, WDTCP improves up to 31.1% (907us for WDTCP, 1316us for UDP at 0.9 load). In summary, WDTCP can significantly reduce latency compared to UDP, with a negligible throughput loss.

2) WDTCP vs. other TCP CCAs: To extensively evaluate the performance of WDTCP, we conduct separate tests in multiple scenarios and compare them with three other TCP CCAs: Vegas, NewReno, and BBR. Firstly, we increase the number of communicating node pairs. More communicating



(a) Throughput under high load. (b) Avg and 95th delay.

Fig. 11: UDP vs. WDTCP



(a) Delay under different pairs. (b) Throughput under packet loss

Fig. 12: Performance comparison with other CCAs

nodes increase the competition for physical links so the sending opportunities for each pair will be reduced proportionally. The results can be seen in Fig 12(a). TCP-NewReno and WDTCP achieve similar results (Up to 15% difference under 4 pairs), with lower average delay than BBR and Vegas. More communicating nodes will increase channel competition and WDTCP can quickly sense the channel changes.

Next, we evaluate the loss tolerance. We drop packets at a 3% loss rate and test the performance under different link capabilities. Figure 12(b) shows using WDTCP can obtain high throughput (Up to 53% higher than NewReno under HeMcs11). The reason why BBR obtains poor performance is that it hardly estimates the bandwidth accurately in a weak network environment. We also compare it with the state-of-the-art scheme TACK [8]. In its test scenario, the throughput of WDTCP is 743Mbps, which is 33.6% higher than TACK (556Mbps). Although TACK has reduced the number of ACKs, overhead still exists under high throughput.

VII. CONCLUSION

To the best of our knowledge, this is the first work to give a full protocol design of cross-layer transmission control over Wi-Fi Direct. We propose WDTCP, which works in a cross-layer paradigm and tightly couples the transport and link layers via the AD and ECC components. Our evaluation results demonstrate that WDTCP significantly improves data

throughput, which can be attributed to the design of AD. Meanwhile, WDTCP still achieves low-latency transmission. This can be attributed to the design of ECC. WDTCP shows potential in the one-hop WLAN scenarios.

REFERENCES

[1] J. H. Lee, M.-S. Park, and S. C. Shah, "Wi-fi direct based mobile ad hoc network," in *IEEE ICCCS*, 2017, pp. 116–120.

[2] K. Xu, Y. Qu, and K. Yang, "A tutorial on the internet of things: from a heterogeneous network integration perspective," *IEEE network*, vol. 30, no. 2, pp. 102–108, 2016.

[3] "Wi-fi direct and file transfer windows 10/11: How to send files from pc to pc/phone to pc. (2022). https://www.easeus.com/pc-transfer/wifi-direct-and-file-transfer-windows-10-11.html/." 2022.

[4] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, "Device-to-device communications with wi-fi direct: overview and experimentation," *IEEE wireless communications*, vol. 20, no. 3, pp. 96–104, 2013.

[5] "The screen mirroring or wi-fi direct compatible device doesn't detect my tv. (2022). https://www.sony.com/electronics/support/articles/00070729/." 2022.

[6] A. Natani, J. Jakilinki, M. Mohsin, and V. Sharma, "Tcp for wireless networks," *Computer Journal of ACM SIGCOMM*, vol. 26, no. 5, pp. 163–243, 2001.

[7] A. M. Al-Jubari, M. Othman, B. Mohd Ali, and N. A. W. Abdul Hamid, "Tcp performance in multi-hop wireless ad hoc networks: challenges and solution," *EURASIP Journal on Wireless Communications and Networking*, vol. 2011, no. 1, pp. 1–25, 2011.

[8] T. Li, K. Zheng, K. Xu, R. A. Jadhav, T. Xiong, K. Winstein, and K. Tan, "Tack: Improving wireless transport performance by taming acknowledgments," in *ACM SIGCOMM*, 2020, pp. 15–30.

[9] T. Li, K. Zheng, and K. Xu, "Acknowledgment on demand for transport control," *IEEE Internet Computing*, vol. 25, no. 2, pp. 109–115, 2021.

[10] S. Ha, I. Rhee, and L. Xu, "Cubic: a new tcp-friendly high-speed tcp variant," *ACM SIGOPS operating systems review*, vol. 42, no. 5, pp. 64–74, 2008.

[11] N. Parvez, A. Mahanti, and C. Williamson, "An analytic throughput model for tcp newreno," *IEEE/ACM Transactions on Networking*, vol. 18, no. 2, pp. 448–461, 2009.

[12] L. S. Brakmo, S. W. O'Malley, and L. L. Peterson, "Tcp vegas: New techniques for congestion detection and avoidance," in *ACM SIGCOMM*, 1994, pp. 24–35.

[13] B. Fu, Y. Xiao, H. Deng, and H. Zeng, "A survey of cross-layer designs in wireless networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 110–126, 2013.

[14] L. Salameh, A. Zhushi, M. Handley, K. Jamieson, and B. Karp, "Hack: Hierarchical acks for efficient wireless medium utilization," in *USENIX ATC*, 2014, pp. 359–370.

[15] Y. Xie, F. Yi, and K. Jamieson, "Pbe-cc: Congestion control via endpoint-centric, physical-layer bandwidth measurements," in *ACM SIGCOMM*, 2020, pp. 451–464.

[16] Y. Gu, F. Ren, Y. Ji, and J. Li, "The evolution of sink mobility management in wireless sensor networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 507–524, 2015.

[17] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "Xors in the air: Practical wireless network coding," in *ACM SIGCOMM*, 2006, pp. 243–254.

[18] Y. Zaki, T. Pötsch, J. Chen, L. Subramanian, and C. Görg, "Adaptive congestion control for unpredictable cellular networks," in *ACM SIGCOMM*, 2015, pp. 509–522.

[19] K. Winstein, A. Sivaraman, and H. Balakrishnan, "Stochastic forecasts achieve high throughput and low delay over cellular networks," in *USENIX NSDI*, 2013, pp. 459–471.

[20] R.-S. Cheng and H.-T. Lin, "A cross-layer design for tcp end-to-end performance improvement in multi-hop wireless networks," *Computer Communications*, vol. 31, no. 14, pp. 3145–3152, 2008.

[21] S. Thangam and E. Kirubakaran, "A survey on cross-layer based approach for improving tcp performance in multi hop mobile adhoc networks," in *IEEE ICETC*, 2009, pp. 294–298.

[22] Y. Liu, X. Wang, G. Boudreau, A. B. Sediq, and H. Abou-Zeid, "A multi-dimensional intelligent multiple access technique for 5g beyond and 6g wireless networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1308–1320, 2020.

[23] W. Sun, C. Yang, S. Jin, and S. Choi, "Listen channel randomization for faster wi-fi direct device discovery," in *IEEE INFOCOM*, 2016, pp. 1–9.

[24] Z. Wang, F. Li, X. Wang, T. Li, and T. Hong, "A wifi-direct based local communication system," in *IEEE IWQoS*, 2018, pp. 1–6.

[25] "Wi-fi alliance. (2023). https://www.wi-fi.org/," 2023.

[26] "Huawei share. (2023). https://consumer.huawei.com/en/support/content/en-us15909309/." 2023.

[27] "ios: What wireless technology does airdrop use? (2023). https://www.quora.com/ios-what-wireless-technology-does-airdrop-use/." 2023.

[28] "What is the mcs index? how to read and measure it? (2021). https://www.accessagility.com/blog/what-is-mcs-index-for-windows/." 2021.

[29] "Bbr congestion control work at google ietf 101 update. (2018). https://datatracker.ietf.org/meeting/101/materials/slides-101-iccrg-an-update-on-bbr-work-at-google-00/." 2018.

[30] A. Seyedolhosseini, N. Masoumi, and M. Modarressi, "Performance improvement of zigbee networks in coexistence of wi-fi signals," in *ACM ICICM*, 2017, pp. 45–49.

[31] L. Li, K. Xu, T. Li, K. Zheng, C. Peng, D. Wang, X. Wang, M. Shen, and R. Mijumbi, "A measurement study on multi-path tcp with multiple cellular carriers on high speed rails," in *ACM SIGCOMM*, 2018, pp. 161–175.

[32] T. Li, W. Liu, X. Ma, S. Zhu, J. Cao, S. Liu, T. Zhang, Y. Zhu, B. Wu, and K. Xu, "Art: Adaptive retransmission for wide-area loss recovery in the wild," in *IEEE ICNP*. IEEE, 2023, pp. 1–11.

[33] M. Al-Sadi, R. Di Pietro, F. Lombardi, and M. Signorini, "Lento: Unpredictable latency-based continuous authentication for network intensive iot environments," *Future Generation Computer Systems*, vol. 139, pp. 151–166, 2023.

[34] F. Gabriel, J. Acevedo, and F. H. Fitzek, "Network coding on wireless multipath for tactile internet with latency and resilience requirements," in *IEEE GLOBECOM*, 2018, pp. 1–6.

[35] W. Shudong and M. Higgins, "Limitations of mobile phone learning," in *IEEE WMTE*, 2005, pp. 3–pp.

[36] P. Chatzimisios, A. C. Boucouvalas, and V. Vitsas, "Ieee 802.11 packet delay-a finite retry limit analysis," in *GLOBECOM'03. IEEE global telecommunications conference (IEEE Cat. No. 03CH37489)*, vol. 2. IEEE, 2003, pp. 950–954.

[37] I. C. S. L. S. Committee *et al.*, "Ieee standard for information technology-telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements part 11: Wireless lan medium access control (mac) and physical layer (phy) specifications," *IEEE Std 802.11ˆ*, 2007.

[38] Y. Tay and K. C. Chua, "A capacity analysis for the ieee 802.11 mac protocol," *Wireless networks*, vol. 7, no. 2, pp. 159–171, 2001.

[39] M. Amjad, L. Musavian, and M. H. Rehmani, "Effective capacity in wireless networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3007–3038, 2019.

[40] T. Vanhatupa, "Wi-fi capacity analysis for 802.11 ac and 802.11 n: Theory & practice," *Ekahau Inc*, 2013.

[41] M. Takai, J. Martin, and R. Bagrodia, "Effects of wireless physical layer modeling in mobile ad hoc networks," in *ACM MobiHoc*, 2001, pp. 87–94.

[42] G. L. Stuber, J. R. Barry, S. W. Mclaughlin, Y. Li, M. A. Ingram, and T. G. Pratt, "Broadband mimo-ofdm wireless communications," *Proceedings of the IEEE*, vol. 92, no. 2, pp. 271–294, 2004.

[43] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, "Bbr: Congestion-based congestion control: Measuring bottleneck bandwidth and round-trip propagation time," *Queue*, vol. 14, no. 5, pp. 20–53, 2016.

[44] D. Schoonwinkel, "Practical measurements of wi-fi direct in content sharing, social gaming android applications," Ph.D. dissertation, Stellenbosch: Stellenbosch University, 2016.

[45] N. Afrin, J. Brown, and J. Y. Khan, "Design of a buffer and channel adaptive lte semi-persistent scheduler for m2m communications," in *IEEE ICC*, 2015, pp. 5821–5826.

[46] R. Jain, *The art of computer systems performance analysis: techniques for experimental design, measurement, simulation, and modeling*. John Wiley & Sons, 1990.