USENIX

THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

BLADE: Adaptive Wi-Fi Contention Control for Next-Generation Real-Time Communication

Fengqian Guo, *Tencent;* Yuhan Zhou, *Peking University and Tencent;* Longwei Jiang and Congcong Miao, *Tencent;* Yuxin Liu, *University at Buffalo, SUNY;* Chenren Xu, *Peking University;* Hancheng Lu, *Institute of Artificial Intelligence, China;* Chang Wen Chen, *The Hong Kong Polytechnic University;* Yaxiong Xie, *University at Buffalo, SUNY;* Honghao Liu, *Tencent*

https://www.usenix.org/conference/nsdi26/presentation/guo-fengqian

This paper is included in the Proceedings of the 23rd USENIX Symposium on Networked Systems Design and Implementation.

May 4–6, 2026 • Renton, WA, USA

ISBN 978-1-939133-54-0

Open access to the Proceedings of the 23rd USENIX Symposium on Networked Systems Design and Implementation is sponsored by

جامعة الملك عبدالله للعلوم والتقنية

King Abdullah University of Science and Technology

BLADE: Adaptive Wi-Fi Contention Control for Next-Generation Real-Time Communication

Fengqian Guo[1*] Yuhan Zhou[2 1*] Longwei Jiang[1] Congcong Miao[1] Yuxin Liu[3]
Chenren Xu[2] Hancheng Lu[4] Chang Wen Chen[5] Yaxiong Xie[3†] Honghao Liu[1†]

[1]*Tencent* [2]*Peking University* [3]*University at Buffalo, SUNY*
[4]*Institute of Artificial Intelligence, China* [5]*The Hong Kong Polytechnic University*

Abstract

Next-generation real-time communication (NGRTC) applications, such as cloud gaming and XR, demand consistently ultra-low latency. However, through our first large-scale measurement, we find that despite the deployment of edge servers, dedicated congestion control, and loss recovery mechanisms, cloud gaming users still experience long-tail latency in Wi-Fi networks. We further identify that Wi-Fi last-mile access points (APs) serve as the primary latency bottleneck. Specifically, short-term packet delivery droughts, caused by fundamental limitations in Wi-Fi contention control standards, are the root cause. To address this issue, we propose BLADE, an adaptive contention control algorithm that dynamically adjusts the contention windows (CW) of all Wi-Fi transmitters based on the channel contention level in a fully distributed manner. Our ns3 simulations and real-world evaluations with commercial Wi-Fi APs demonstrate that, compared to standard contention control, BLADE reduces Wi-Fi packet transmission tail latency by over 5× under heavy channel contention and significantly stabilizes MAC throughput while ensuring fast and fair convergence. Consequently, BLADE reduces the video stall rate in cloud gaming by over 90%.

1 Introduction

Emerging next generation real time communication (NGRTC) systems such as cloud gaming [1, 2] and Extended Reality (XR) [3, 4] are revolutionizing how users experience interactive digital content. These applications have been rapidly adopted across both entertainment and business, with the global cloud gaming market alone growing from $1,286.6 million in 2022 to a projected $13.6 billion by 2028 [5]. Such next-generation real-time streaming applications require both *high bandwidth* (*e.g.,* ~30 Mbps for cloud gaming [6] and ~50 Mbps for XR [7]) and *consistently low latency* to maintain their interactive nature and deliver immersive user experiences [8, 9].

Long-tail Latency. Except average latency and throughput, NGRTC are extremely sensitive to long-tail latency: even a single latency spike, *i.e.*, latency larger than 200 ms, can trigger a video stall, catastrophically disrupting the user's immersive experience [8–13]. The stakes are high; prior work has shown that a mere 0.5% increase in stall rate can reduce user retention time by a third [8, 9]. Consequently, the viability of this burgeoning multi-billion dollar market hinges on delivering data with near-perfect punctuality.

To prevent these disruptive stalls, the underlying network must provide highly predictable, timely delivery. With a video stall defined as any frame delivery taking longer than 200 ms, NG-RTC applications implicitly demand that the network deliver nearly every frame within this strict budget; even failures at the 99.99th percentile can severely degrade the user experience. This imposes a stringent requirement for predictable network performance.

The Internet's most prevalent last-hop technology, Wi-Fi, is theoretically incapable of providing this guarantee. Unlike cellular networks, which leverage centralized scheduling to allocate resources and manage latency [14, 15], Wi-Fi's design is fundamentally distributed and uncoordinated. Its reliance on a contention-based channel access protocol (CSMA/CA) means that devices must independently compete for transmission opportunities without any central control. This lack of coordination makes it impossible to guarantee timely packet delivery, as latency can vary dramatically based on the instantaneous level of channel contention.

Our large-scale measurement study on the Tencent START cloud gaming platform confirms this theoretical limitation (§3.1). The study, which analyzed 336 million video frames from 200 commercial Wi-Fi access points deployed nationwide, reveals that while the wired portion of the network (from server to access point) maintains low latency, staying below 200 ms even at the 99.99th percentile, the total end-to-end latency can exceed 1000ms when the wireless last hop is included. This empirical evidence dictates that the solution must reside at the Wi-Fi last hop. This is not a problem that traditional end-to-end congestion control can solve, as congestion control only mitigates queuing delay rather than reducing the brief, intermittent sudden jitters inherent to the Wi-Fi last hop. The catastrophic latency spikes are brief, intermittent, and localized entirely within the Wi-Fi last hop. Therefore, resolving the random jitters of over-the-air transmission is critical for NGRTC. Our research focus is on the core cause

*equal contribution
†Corresponding authors.

of video stalls: the Wi-Fi last hop. We address this by designing a deployable Media Access Control (MAC)-layer contention-control algorithm that reduces the tail latency of the last hop and enhances the smoothness of application-layer transmission.

Packet-Delivery Drought and its Root Cause. Drilling down into the Wi-Fi bottleneck, our measurements reveal that these latency spikes are caused by a specific, recurring failure mode we term a *packet-delivery drought*: a 200 ms interval during which an access point fails to deliver *a single packet* to a user. This MAC-layer phenomenon is the direct cause of application-layer failure: video stalls. Our central empirical finding is that 86.19% of all video stalls are directly correlated with the occurrence of at least one such drought, establishing a near one-to-one mapping between the two.

The root cause of these droughts is not slow physical transmission or a lack of channel capacity. Our measurements confirm that the time spent on physical packet transmission (PHY TX) is consistently brief, with a 99.99th-percentile delay below 5 ms. In stark contrast, the contention interval, *i.e.*, the time a device spends waiting for channel access, exhibits an alarming heavy tail, exceeding 200 ms at the 99.99th percentile. This delay originates from a fundamental flaw in the IEEE 802.11 CSMA/CA protocol: *short-term unfairness driven by its exponential backoff mechanism*. Specifically, after a collision, a device doubles its contention window (CW), creating a temporary but severe priority asymmetry. Other devices with smaller CWs can repeatedly seize the channel while the device with the large CW is forced to wait, its backoff counter perpetually frozen. In congested environments, these interruptions can *extend a simple backoff from milliseconds to hundreds of milliseconds*, starving the device of access and creating a packet-delivery drought. This is a failure of micro-fairness, not aggregate channel efficiency.

A conventional approach is to leverage existing Wi-Fi Quality of Service (QoS) mechanisms—such as the priority queues of the Enhanced Distributed Channel Access (EDCA) mechanism defined in the IEEE 802.11e standard. On the one hand, encrypted traffic has become the dominant form of Internet traffic [16], making the identification of traffic for specific priority levels extremely challenging. On the other hand, in dense network environments, concurrent contention for the channel by multiple high-priority traffic flows merely intensifies channel competition, leading to more frequent packet collisions and thereby exacerbating the tail latency issues that such QoS mechanisms were originally designed to mitigate. This demonstrates that a simple priority scheme is insufficient. What is needed is a cooperative mechanism that allows all devices to adapt to the actual level of channel contention.

Solution: Predictable and Cooperative Contention. To eliminate these droughts, we must replace Wi-Fi's flawed signaling with a mechanism that enables predictable, cooperative behavior. We present BLADE, an adaptive contention control algorithm that fundamentally changes how devices perceive and react to network congestion. The critical flaw in the standard protocol is its reliance on a *local and reactive signal*: a collision. While all devices listen before talk using clear channel assessment (CCA), they only adjust their behavior aggressively after their own transmission fails. This signal is local: uninvolved devices remain oblivious to the contention event; and reactive: addressing congestion only after it has already caused a failure. This leads to uncoordinated responses where some devices are forced to wait while others, with smaller contention windows, continue to seize the channel, creating the priority asymmetries that cause droughts.

BLADE solves this by deriving a universal and proactive signal from the same CCA mechanism. Instead of waiting for a personal failure, each device continuously measures the *microscopic access rate* (MAR): the ratio of successful transmission events (from any device) to the number of idle time slots it observes. Because the protocol forces every device to defer to any ongoing transmission, devices within the same carrier-sense domain typically observe consistent busy/idle slot dynamics. (Hidden terminals and partial visibility can violate this assumption; we make this explicit and discuss mitigation via RTS/CTS in §4.2.1 and §H.) This provides a consistent, shared, and quantitative measure of the channel's current contention level. By shifting from a local, reactive signal to a universal, proactive one, BLADE enables all devices to act cooperatively. They adjust their contention windows based on a shared understanding of network congestion, preventing the short-term unfairness that causes packet-delivery droughts in the first place. BLADE is a MAC-layer transmitter-side mechanism. In our primary, downlink-dominated cloud gaming setting, deploying BLADE on APs (the dominant transmitters) already addresses long-tail contention among neighboring APs and does not require client STA modifications; when uplink traffic is significant, an AP can optionally advertise contention parameters via standards-compliant EDCA parameter sets, or STAs can run BLADE locally.

BLADE achieves this cooperative behavior through two core mechanisms. First, it introduces the universally observable contention signal: MAR. Second, it employs a *hybrid increase multiplicative decrease* (HIMD) policy that uses MAR as feedback, enabling co-channel devices to collectively and dynamically adapt their contention windows to match competition level. This allows the network to converge on fair and efficient operation without explicit coordination, proactively preventing the priority asymmetries that cause droughts.

We evaluate BLADE through extensive real-world experiments with commercial Wi-Fi APs and ns3 simulations. The results demonstrate that BLADE directly remedies the root cause of tail latency. Compared to the standard contention control, BLADE reduces Wi-Fi packet transmission tail latency by over 5× under heavy channel contention. This MAC-layer improvement translates directly to application-level benefits: for cloud gaming, BLADE reduces the 99th-percentile video frame delivery latency to $\leq$0.5× the baseline and, conse-

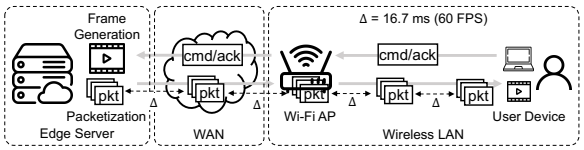


Figure 1: The system architecture of next-generation real-time streaming over wireless LAN.

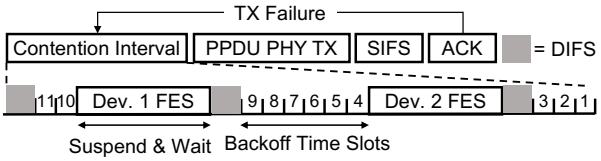


Figure 2: Wi-Fi Frame exchange sequence.

quently, cuts the video stall rate by over 90%.

Contribution. This paper makes the following contributions:

- We conduct a large-scale measurement of a commercial cloud gaming service and identify that packet-delivery droughts in the Wi-Fi last hop, caused by fundamental limitations in standard contention control, are the root cause of high tail latency for NGRTC applications.
- We design and implement BLADE, an adaptive contention control algorithm that dynamically and cooperatively adjusts the contention windows of all transmitters based on a novel, universally observable contention signal.
- We evaluate BLADE using both simulations and commercial Wi-Fi APs, demonstrating that it significantly reduces Wi-Fi packet transmission latency and stabilizes throughput, ultimately reducing the video stall rate by over 90%.

Ethical claim. All user data collected in this work are obtained with explicit permission from the users and are anonymized to protect their privacy. This work does not raise any ethical concerns and conforms to the IRB policies of the authors' institutions.

2 Background

To understand the latency challenges facing NGRTC applications, this section provides essential background. We first describe the architecture of these systems and their strict performance requirements. We then examine the details of Wi-Fi's contention-based channel access, which is the root of the performance bottlenecks we address.

2.1 Next-Generation RTC

System Architecture. The inherent contradiction between computing power demands and terminal portability has given rise to the core technical paradigm of computation and interaction decoupling — computationally intensive tasks are processed in the cloud, while terminals focus on low-latency local interaction and content presentation. Typical applications such as cloud extended reality, AI smart glasses, and cloud gaming are all built on this paradigm. Compared with traditional Real-Time Communication (RTC) scenarios like live streaming and video conferencing, these immersive interactive applications impose much more stringent performance requirements on network transmission [8, 17, 18]: the end-to-end latency needs to be reduced from hundreds of milliseconds to tens of milliseconds, and the bit-rate must be increased from several megabits per second to tens of megabits per second. We define such dedicated network transmission demands for immersive real-time interaction as Next-Generation Real-Time Communication (NGRTC). Based on our analysis at Tencent START cloud gaming service, [1]. A typical NGRTC over WLAN system comprises four key components: the cloud server, WAN, Wi-Fi Access Point (AP), and user device, as illustrated in Fig. 1. The cloud server generates video frames at a fixed frame rate, and each frame is packetized into multiple packets for network transmission. For example, at 60 FPS, a new frame is generated and transmitted every 16.7 ms. These packets traverse the WAN to reach the Wi-Fi AP, which then delivers them wirelessly to the user's device. The system operates bidirectionally - upon receiving video frames, the user device sends acknowledgments (ACKs) along with interactive commands (*e.g.,* character movement or action triggers in mobile games) back to the cloud server. These user inputs then influence the generation of subsequent video frames.

QoE Requirements on WLAN. The QoE of NGRTC critically depends on two key parameters: video quality [8] and interaction smoothness [8–10, 19, 20]. For better video quality, these applications stream at much *higher bitrates* (*e.g.,* over 30 Mbps for cloud gaming [8] and over 200 Mbps for VR [21]) compared to traditional RTC applications. For smooth interactions, they require a *higher frame rate* (*i.e.,* 60 to 144 FPS) and demand *consistently low video frame delivery latency*. Specifically, *tail latency* is particularly crucial: elevated tail latency–even at 99.99th percentile–can directly cause frequent video freezes and stalls [8, 9], significantly degrading user experience: a recent study [9] has shown that even a minor 0.5% increase in stall rate leads to a dramatic 33% reduction in user retention time. Thus, although Wi-Fi 6/7 offers theoretical rates of 9.6 Gbps and 46 Gbps, far exceeding NGRTC's bitrate requirements, NGRTC's sensitivity to tail latency imposes higher demands on the real-time performance and stability of wireless network transmissions.

2.2 WLAN Channel Access

In this section, we introduce Wi-Fi's contention-based channel access and packet transmission procedures.

Channel via CSMA/CA. Wi-Fi leverages *carrier sense multiple access/collision avoidance* (CSMA/CA) for channel access, which requires a device to monitor channel activity before packet transmission. Specifically, as illustrated in Fig. 2, a device must detect the channel as idle for *B backoff slots* before initiating transmission. The value of *B* is randomly chosen from the range [0, CW] upon each transmission, where

[1] We infer access type from the client's active network interface at session start and exclude sessions where access type is unknown.

CW is the *contention window* of the device. If the device detects an ongoing transmission during its countdown from B, it suspends the countdown and resumes only after detecting the channel as idle for a *DCF* [2] *interframe space* (DIFS) interval. Upon successful completion of the B slot countdown, the device gains channel access for transmission.

Channel Contention Interval. We define the *contention interval* as the period starting from DIFS until the successful completion of B backoff slots countdown. During one device's contention interval, other devices may gain channel access first, as illustrated in Fig. 2. Consequently, the duration of a contention interval is determined by two factors: the initial number of backoff slots (B) and the number of channel access instances obtained by competing devices.

Wi-Fi Packet Transmission Procedure. After gaining channel access through CSMA/CA, the device encapsulates packets into *PLCP Protocol Data Units* (PPDUs) with radio headers and proceeds with packet transmission during the *PHY TX* period shown in Fig. 2. Upon successful PPDU reception, the receiver must acknowledge by sending an ACK frame after a *short inter-frame space* (SIFS) interval. If a transmission fails (*i.e.,* no ACK or a NACK is received), the sender triggers a retransmission and re-gains channel access through CSMA/CA. From the initial DIFS to the final ACK, it is defined as a *frame exchange sequence* (FES) for a PPDU.

Wi-Fi MAC Throughput Analysis. Wi-Fi MAC throughput is determined by three key components: *i)* PHY transmission rate, which dictates how quickly data packets can be transmitted over the air; *ii)* Channel access overhead, including variable contention intervals and fixed intervals like DIFS, SIFS, and ACK; *iii)* Transmission failure rate, failures are primarily caused by poor signal strength or signal collisions, where multiple devices attempt to transmit simultaneously.

2.3 Predictability VS. Efficiency in CSMA

It is important to distinguish the problem of tail latency from the well-studied issue of CSMA efficiency. CSMA efficiency is typically defined as the ratio of airtime used for successful data payload transmission to the total airtime consumed by a full frame exchange sequence, which includes fixed overheads like the contention interval, DIFS, SIFS, and the ACK frame. A long-standing challenge in Wi-Fi has been that as physical data rates increased, the time to transmit a packet shrank, while these overheads remained constant, causing a decline in overall channel efficiency.

This problem has been the subject of extensive research in the past two decades. Representative examples include fine-grained or frequency-domain contention to shrink time-domain overheads [23, 24], explicit collision notification to curtail wasted airtime and hidden-terminal losses [25], and hybrid centralized–distributed coordination to exploit controller visibility while retaining CSMA agility [26]. Earlier algorithmic tuning of contention parameters likewise sought high throughput and fairness under CSMA [27]. This problem is now largely mitigated in modern Wi-Fi standards by highly effective solutions like frame aggregation (*e.g.*, A-MPDU). By allowing multiple packets to be transmitted after a single contention event, aggregation amortizes the overhead cost and significantly improves system throughput.

This paper, however, does not aim to solve the general CSMA efficiency problem. Instead, we focus on a distinct but equally critical issue: the opportunistic and severe inflation of the contention interval for specific packets. While low efficiency is a systemic issue affecting average throughput, the problem we address is transient and statistical. The long-term throughput and average latency for a user can be perfectly acceptable, yet the user's experience can be ruined by intermittent packet-delivery droughts where the contention window for a single video frame inflates to an extreme value. Efficiency is a problem of averages; contention-driven tail latency is a problem of outliers, and for NGRTC applications, these outliers are catastrophic.

3 Measurement and Motivation

In this section, we build the case for our proposed solution by first identifying and then diagnosing the core performance problem for NGRTC applications in today's Wi-Fi networks.

3.1 Large-Scale Online Measurement

In this section, we present the first large-scale measurement study of Wi-Fi performance for NGRTC, revealing critical limitations in current Wi-Fi APs' ability to support these demanding workloads.

While it is well known that CSMA/CA contention introduces variable per-packet delay, the application-level implications for modern high-bitrate interactive streaming—and the concrete failure mode that triggers stalls—have been unclear. Our measurement contributes three pieces of evidence: (i) quantified stall-rate tails under Wi-Fi versus wired access, (ii) latency decomposition showing that the Wi-Fi last hop dominates even when WAN RTT is low and stable, and (iii) a near one-to-one correlation between 200 ms packet-delivery droughts and video stalls. These findings motivate a link-layer contention-control mechanism that targets micro-level access fairness and bounds last-hop tail latency.

3.1.1 Testbed and Data Collection Scheme

Testbed. To understand how Wi-Fi last-hop performance impacts next-generation real-time streaming applications, we conducted an extensive measurement study through Tencent START cloud gaming platform. This platform delivers high-quality interactive gaming content, streaming 1080p to 4K video at 60-144 FPS with bitrates around 50 Mbps. Our measurement infrastructure consists of 200 commercial Wi-Fi access points distributed to volunteer users nationwide. To reduce and stabilize server-side queuing delay (so that last-hop effects are more visible), we deployed Pudica [8] on the cloud-gaming servers. Pudica enables near-zero queu-

[2] DCF means distributed coordinate function in IEEE 802.11 [22].

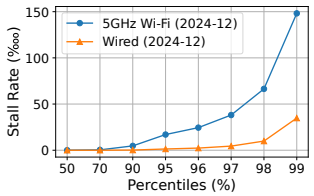


Figure 3: Stall rate percentiles in Dec. 2024.

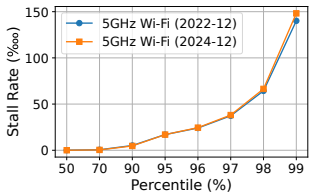


Figure 4: Stall rate for 5 GHz Wi-Fi in Dec. 2022 and 2024.

ing delay, emerging as the state-of-the-art congestion control algorithm tailored for low-latency demands.

Data Collection. To collect comprehensive network performance data, we instrumented the WNIC driver on our Wi-Fi APs to report essential channel status, MAC layer metrics, and PHY layer parameters. The AP also records the successfully transmitted packets within each 200 ms interval, providing direct insight into wireless channel contention. Along with traditional metrics like RSSI, transmission delay, packet loss, and channel properties, the APs report these measurements every 200 ms. This data collection scheme allows us to measure server-to-router RTT and distinguish between wired and wireless latency issues. Our server also collects transport layer statistics including frame-level RTT, packet loss, and jitter. Concretely, the AP periodically measures a server↔AP RTT (every 200 ms) over the control channel and reports it with the same granularity as MAC/PHY metrics. The server separately obtains per-frame end-to-end RTT from the cloud-gaming feedback path. We align the two by time (no clock synchronization is needed because both are RTTs). Over one year, we gathered data from 336 million video frames—representing the first large-scale study of wireless last-hop performance for real-time streaming.

3.1.2 Measurement Results

High Video Stall Rate. Fig. 3 shows the video stall rate[3] percentiles for cloud gaming users in December 2024 across different networks. We report stall rate as stalls per 10,000 frames ($\times 10^{-4}$), so values above 100 correspond to more than 1% of frames stalling. 5 GHz Wi-Fi exhibits significantly higher tail latency compared to wired networks, indicating the superior stability of wired connections. Fig. 4 compares the stall-rate percentiles for 5 GHz Wi-Fi sessions from two matched one-month snapshots (Dec. 2022 vs. Dec. 2024) under the same stall definition. The similarity indicates that, even as Wi-Fi hardware evolves, tail stalls driven by CSMA/CA contention remain a dominant factor in dense environments.

Wi-Fi Last-Mile Causes High Video Stall Rate. Our measurement campaign reveals the root cause of these video stalls: the Wi-Fi last hop acts as a critical performance bottleneck. By analyzing each video frame's end-to-end path, we find striking differences between wired and wireless segments. As

[3]We define a video stall as occurring when end-to-end frame delivery latency exceeds 200 ms. This metric is based on user QoE feedback and has been adopted by many previous studies in NGRTC [8, 9, 11].

Range	Probability (%)	Range	Probability (%)
0	86.19	5	0.78
1	0.29	[6,10)	2.55
2	0.39	[10,20)	2.86
3	0.36	[20,50)	2.46
4	0.29	(50,∞)	3.82

Table 1: The distribution of the number of packets transmitted by the Wi-Fi router within 200 milliseconds, when downstream long-tail latency occurs (with the absence of wired network issues confirmed).

shown in Fig. 5, the wired portion (server to AP) maintains consistently low latency, staying below 200 ms even at the 99.99th percentile. However, when including the wireless last hop (AP to user), the total latency can exceed 1000 ms. To precisely quantify this impact, we decomposed each frame's delivery time into wired and wireless components. Fig. 6 reveals that the wireless segment contributes disproportionately to total latency, with its share growing dramatically as delivery times increase. This finding is particularly concerning because it shows that even with state-of-the-art WAN congestion control, the wireless last hop remains the primary obstacle to reliable real-time streaming.

Packet Delivery Droughts: Root Cause of Frame Stalls. To identify the root cause of Wi-Fi last-hop delays, we analyzed the correlation between stalled video frames and successful packet transmissions. For each frame with a high end-to-end delay (server to client), we examined the number of successfully transmitted packets within each 200 ms window of the frame's transmission. To isolate Wi-Fi-induced stalls, we focused on frames where server-to-client latency exceeded 200 ms while server-to-router RTT remained below 50 ms—effectively filtering out stalls caused by wired network issues.

Table 1 reveals a striking pattern: in 86.19% of these stalled frames, the router failed to successfully transmit even a single packet during at least one 200 ms interval, despite potentially having transmission opportunities. This near one-to-one correspondence between packet delivery droughts and frame stalls suggests a fundamental issue in Wi-Fi's channel access mechanism, where either transmission opportunities are not obtained or packets fail to be delivered even when opportunities are granted. In contrast, we calculate the PHY transmission delay of PPDUs and present its distribution in Fig. 7. Once PPDUs are granted transmission opportunities, the actual transmission completes quickly, with 92.7% finishing within 3.5 ms and a maximum delay of 7.5 ms.

To further understand why AP fails to deliver packets, we investigated the relationship between packet delivery and channel contention. We define the channel contention rate as the proportion of airtime occupied by other transmitters within each 200 ms interval (longer airtime by others indicating higher contention). Fig. 8 shows that the probability of zero packet deliveries rises dramatically with increased channel contention—when contention exceeds 80%, the prob-

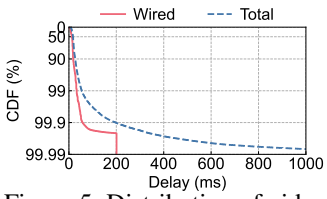


Figure 5: Distribution of video frame latency in cloud gaming.

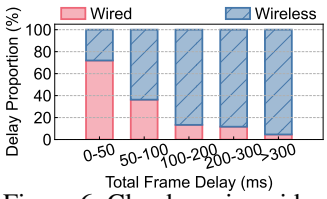


Figure 6: Cloud gaming video frame latency decomposition.

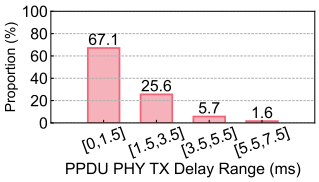


Figure 7: Distribution of Wi-Fi PHY transmission delay.

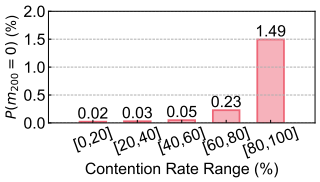


Figure 8: The distribution of transmission opportunity.

AP Num.	Session Num.	Stall Rate (%)
2	52349	0.08
4	25624	0.17
6	14414	0.42
≥ 8	7976	1.34

Table 2: Relations between video stall rate of Wi-Fi sessions and Wi-Fi AP numbers in the environment from our online cloud gaming platform for 8 weeks.

ability of a complete delivery drought is 74.5 times higher than under 20% contention.

We further validated this relationship through an 8-week field study where, with user consent, we monitored the number of nearby Wi-Fi APs as a proxy for potential channel contention. Table 2 demonstrates that video stall rates, particularly at the tail, increase systematically with the number of surrounding APs. These findings establish that frame stalls primarily occur when routers experience packet delivery droughts during periods of intensive channel contention. Importantly, a single 200 ms delivery drought already crosses the stall threshold used by the application, so mitigating micro-level droughts is directly reflected in lower stall rate and better user QoE.

3.2 Mechanism Behind Packet Delivery Droughts

Our online measurements establish that delivery droughts concentrate under high contention, but they do not expose the packet-level dynamics that create 100–200 ms gaps. We therefore complement them with ns-3 simulations and controlled experiments with commercial Wi-Fi APs. Across both settings, we find that (i) collisions increase retransmissions, and (ii) each retransmission triggers binary exponential backoff whose countdown is repeatedly frozen under a busy channel, stretching the *effective* contention interval from sub-millisecond to hundreds of milliseconds. We report the full methodology and supporting figures in §D. These dynamics point to a deeper limitation: *802.11's contention control is collision-driven and purely reactive*, which we summarize next.

3.2.1 Root Cause: Collision-Driven Reactive Contention

The fundamental issue lies in 802.11's reactive approach to contention control. As detailed in §D, the long gaps are dominated by collision-driven retransmissions and prolonged countdown freezes, rather than PHY transmission time. Current CSMA/CA mechanism has two critical limitations that lead to extended packet delivery times. First, the protocol always initializes transmission with a small contention window (CW_{min}), regardless of network contention levels. In dense networks with high contention, this approach inevitably leads to frequent collisions—multiple devices are likely to select similar small backoff values. A more effective approach would be to proactively adjust the initial window size based on observed network contention, starting with larger windows when contention is high.

Second, the protocol creates unfair channel access after collisions. When a device experiences a collision, it doubles its contention window, while devices without recent collisions maintain small windows. This creates a problematic asymmetry: devices with larger windows must count down through more slots, making them more likely to be interrupted by transmissions from devices with smaller windows. Each interruption forces the device to pause its countdown until the channel becomes idle again. In dense networks, these interruptions can extend a simple backoff period from milliseconds to hundreds of milliseconds.

This reactive, device-by-device approach means the system never achieves a coordinated response to network contention. Instead, devices independently adjust their windows based only on their own collision experiences, leading to persistent unfairness and inefficient channel utilization. A better approach would be to maintain similar window sizes across all devices based on overall network contention levels, ensuring fair channel access while proactively preventing collisions.

4 BLADE Design

To meet the latency and throughput stability requirements of NGRTC in Wi-Fi networks, we consider the following two aspects as the most critical: (1) maintaining a low collision probability; (2) ensuring that the CW of different co-channel Wi-Fi devices remains as consistent as possible. Based on these two principles, we design BLADE, which leverages MAR and employs the HIMD approach to adaptively adjust CW, achieving a balance between high throughput and low collision probability, without relying on priority queues in Wi-Fi networks. We first present the design goals of BLADE.

4.1 Design Goals

A practical and effective contention window adjustment mechanism should achieve four key goals:

High Transmission Efficiency. The system must maximize channel utilization by maintaining an optimal collision rate. Since collision recovery (3-5ms) costs significantly more than contention slot time (9μs), we need to balance between avoiding excessive collisions from small contention windows and preventing unnecessary idle periods from large windows.

Fair Channel Access. All devices should reach a consensus on network contention levels and adjust their windows accordingly. This differs from current IEEE mechanisms where devices react individually to their own transmission outcomes, leading to unfair access patterns. Instead, devices should collectively adapt their contention windows based on shared network conditions, ensuring balanced transmission opportunities across the network.

Fast Convergence. The system should rapidly adapt to network changes while maintaining stable operation. When network conditions shift (*e.g.,* , traffic flows joining or leaving), all devices should quickly converge to appropriate window sizes, maintaining both efficiency and fairness in their channel access patterns without oscillating between states.

Minimal Assumptions. The system should be designed without relying on assumptions about user traffic patterns, the number of competing flows, or PPDU PHY transmission duration, as real-world networks are inherently complex. They exhibit unpredictable user traffic, highly dynamic competing transmitters, and varying PHY transmission rates. This contrasts with existing studies on contention window control algorithms [28–32] beyond the IEEE 802.11 standard, which are based on these assumptions.

These goals are particularly challenging because Wi-Fi operates as a fully distributed system where devices must make decisions without explicit coordination.

4.2 Search of a Universal Contention Signal

Requirement. To enable coordinated contention window adjustment across distributed Wi-Fi devices, we need a reliable signal that indicates network contention levels. This signal must satisfy three key requirements: it should be universally observable by all devices, accurately reflect current network competition, and remain stable enough to facilitate consensus. Several candidate signals face fundamental limitations: *i)* Collision-based signals only provide local feedback to involved devices; *ii)* Detecting competing flows requires packet-level decoding at the MAC layer, which is both complex to implement and potentially misleading—flows operate at longer time scales and may be temporarily inactive, making them poor indicators of instantaneous network contention; *iii)* Airtime utilization rate (the fraction of airtime occupied by transmissions) can be deceptive, since high utilization rate may simply caused by large PPDUs from few devices rather than actual competition for channel access.

4.2.1 Proposed Signal: MAR

Definition of MAR. We define the *microscopic access rate* (MAR) as the ratio of transmission opportunities to total available slots in the channel. As shown in Fig. 9, each device monitors both idle slots during its backoff countdown and transmission events in the channel. A transmission event occurs either when the device itself gains channel access or when it detects other devices' transmissions through CCA. Mathematically, MAR is defined as:

$$MAR = \frac{N_{tx}}{N_{tx}+N_{idle}} \quad (1)$$

where N_{tx} is the number of transmission events and N_{idle} is the number of idle slots during backoff countdown. In the example shown in Fig. 9, there are 2 transmission events and 9 idle slots, resulting in a *MAR* of 2/11.

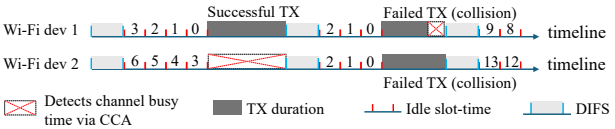


Figure 9: Illustration of MAR. There are 9 idle slot times (in red) and 2 TX durations, the MAR that both device 1 and device 2 detect via CCA is $2/(9+2)$.

MAR: Properties and Advantages. MAR offers three key advantages as a contention signal:

Universal Observability. For devices that can carrier-sense each other (i.e., within the same carrier-sense domain), MAR is consistently observable: when any device transmits, others detect it via CCA and freeze backoff, leading to a shared sequence of transmission events and idle slots. Hidden terminals and partial visibility can violate this assumption; we discuss mitigation via RTS/CTS and empirically validate robustness in §H.

Direct Competition Indicator. MAR directly reflects the intensity of channel competition by measuring the ratio of transmission attempts to available slots. Unlike network utilization or flow counts, MAR captures the actual contention for transmission opportunities, allowing devices to accurately gauge network competition levels.

Predictable Collision Control. When devices maintain MAR at a target threshold through contention window adjustment, collision probability remains stable regardless of the number of competing devices (See §L for proof). This property enables systematic congestion management without requiring knowledge of network size or traffic patterns.

4.3 MAR-Driven Contention Window Control

Problem Statement The core challenge in MAR-driven contention control is to dynamically adjust each transmitter's contention window (CW) to achieve three key objectives. *i)* the system must maintain the observed MAR close to a target value MAR_{tar} to ensure efficient channel utilization; *ii)* all competing transmitters must converge to similar CW values to guarantee fair channel access—significant differences in CW values would give some transmitters unfair advantages in channel competition. *iii)* the system needs to rapidly adapt CW values in response to network changes, converging quickly to optimal settings without oscillation.

These objectives present inherent tensions. Aggressive CW adjustments achieves faster convergence but risks creating temporary unfairness or oscillations. Conservative adjustments provide more stability but may react too slowly to network changes. Additionally, transmitters must achieve these objectives through independent decisions without explicit coordination, as the distributed nature of Wi-Fi networks precludes direct communication between devices.

4.3.1 HIMD-based Contention Window Control

Drawing inspiration from traditional TCP congestion control, we design a *hybrid increase multiplicative decrease* (HIMD) policy for CW adjustment. Note that the "increase/decrease" directions are inverted compared to transport-layer congestion windows: in Wi-Fi, a larger contention window reduces a transmitter's attempt probability, so "increasing CW" makes the transmitter *less* aggressive. Traditional AIMD (additive increase multiplicative decrease) has been proven to achieve fair bandwidth sharing in congestion control. We extend it with a hybrid increase phase that combines both additive and multiplicative components—the additive component ensures steady fairness convergence, while the multiplicative component provides rapid response to severe congestion. This hybrid approach offers better adaptivity than pure AIMD while maintaining its fairness properties. Like AIMD, our HIMD policy increases CW when MAR exceeds MAR_{tar} to reduce channel contention, and decreases CW when MAR is below MAR_{tar} to encourage more transmission attempts. Here, MAR_{tar} is the target microscopic access rate that we regulate to in steady state (default 0.1), and MAR_{max} is an empirical upper bound of MAR under saturated contention (default 0.35), used to normalize/clip the control signal and avoid over-reacting when the channel is nearly fully occupied by transmissions and fixed MAC overheads.

Hybrid Increase. When the observed MAR exceeds the target MAR_{tar}, it indicates excessive contention, leading to more collisions and longer contention intervals. To alleviate this, BLADE increases the contention window CW to yield more transmission opportunities and reduce contention:

$$CW = CW + M_{inc}(\min\{MAR, MAR_{max}\} - MAR_{tar}) + A_{inc} + CW \cdot \max\{0, MAR - MAR_{max}\} \quad (2)$$

Eqn. 2 involves two additive and one multiplicative terms: *i)* Additive term $M_{inc}(\min\{MAR, MAR_{max}\} - MAR_{tar})$ ensures a faster increase in CW when the observed MAR significantly exceeds the target and a slower increase when it is close to the target. We use $M_{inc} = (CW_{max} - CW_{min})/2$ by default; *ii)* Additive term A_{inc} guarantees a minimum increase, promoting fairness among all transmitters' CW values; *iii)* Multiplicative term $CW \cdot \max\{0, MAR - MAR_{max}\}$ is applied to handle extreme contention scenarios. When the observed MAR exceeds MAR_{max}, the channel is considered highly congested and unstable. In response, CW is increased multiplicatively to rapidly reduce contention. Overall, Eqn. 2 behaves as a stable proportional controller on the MAR error within a safe range, while providing (i) a fairness floor via A_{inc} and (ii) an emergency brake via the multiplicative term when the system enters a highly congested regime ($MAR > MAR_{max}$). The default value of MAR_{max} is set to 35%, as our simulation experiments revealed that under the IEEE standard, the MAR tends to rise to approximately 35% with an increasing number of competing Wi-Fi flows.

Multiplicative Decrease. When the observed MAR is below the target MAR_{tar}, it indicates insufficient traffic load, wasting transmission chances and overall bandwidth. To effectively utilize the airtime resource, BLADE should rapidly contend for more transmission chances by decreasing the contention window CW multiplicatively: $CW = \beta \cdot CW, \beta < 1$. Our choice on β value involves two factors: On the one hand, to quickly converge without oscillation, we aim for the observed MAR to increase by $(MAR_{tar} - MAR)/2$ per step. In the converged state, MAR is (approximately) inversely proportional to the converged CW: with attempt probability $\tau \approx \frac{2}{CW+1}$ per transmission chance, $MAR = 1-(1-\tau)^N \approx N\tau \approx \frac{2N}{CW+1}$ for $\tau \ll 1$. Therefore, we use

$$\beta_1 = \frac{MAR}{MAR_{tar} - \frac{MAR_{tar} - MAR}{2}} = \frac{2MAR}{MAR_{tar} + MAR} \quad (3)$$

On the other hand, to accelerate fair convergence, the greater the CW value is, the larger the reduction magnitude should be, therefore, we use

$$\beta_2 = M_{dec} - \frac{(1 - M_{dec})(CW - CW_{min})}{CW_{max} - CW_{min}} \quad (4)$$

where M_{dec} is a minimum decrease factor, with a default value of 0.95. The second term on the right-hand side of Eqn. 4 ensures that Wi-Fi devices with larger CW values experience a greater reduction, thereby speeding up the convergence process. Finally, combining the two considerations, we update the contention window as:

$$CW = \min(\beta_1, \beta_2) \cdot CW \quad (5)$$

Because MAR is a channel-wide consensus signal (all transmitters on the same channel observe the same busy/idle pattern), all nodes react to a common feedback loop. β_1 drives the system toward the fixed point where $MAR \approx MAR_{tar}$, while β_2 contracts CW disparities faster by applying larger reductions to larger CW values. Taking $\min(\beta_1, \beta_2)$ avoids overshooting and reduces oscillation. Together with hard bounds $[CW_{min}, CW_{max}]$, this yields rapid convergence without persistent unfairness.

Target MAR. The target microscopic access rate MAR_{tar} is a critical parameter in our HIMD control policy. Using a standard CSMA/CA throughput model, the throughput-optimal MAR is approximately $MAR_{opt} = \frac{1}{\sqrt{\eta}+1}$, where $\eta = T_c/T_s$ is the collision duration (in slots) relative to an idle backoff slot. In modern Wi-Fi, η is typically large (collisions last tens to hundreds of slots), which places MAR_{opt} in a narrow "safe" band around 0.1. Accordingly, we set $MAR_{tar} = 0.1$ by default, and §6.2.1 shows that BLADE remains robust when MAR_{tar} varies within this band.

Fast Recovery Policy for Collisions. While our HIMD policy ensures stable convergence, random collisions can still occur when multiple transmitters select the same backoff value. To minimize the delay impact of these collisions, we implement a special handling for retransmissions. Upon a transmission failure, instead of the standard IEEE 802.11 approach of doubling CW, we set:

$$CW_{fail} = CW + A_{fail}\ CW = CW_{fail}/2 \quad (6)$$

This temporary CW reduction accelerates retransmission of collided packets while A_{fail} serves as a compensation term. After successful retransmission, we restore CW to CW_{fail} before resuming normal HIMD control. To prevent excessive contention, this halving is applied only to the first retransmission attempt.

5 Implementation

We implement BLADE on Tenda AX12 Pro Wi-Fi APs, accessing the Wi-Fi driver layer to monitor channel activity. Our implementation primarily leverages three hardware counters from the CCA mechanism: *TX_time*: duration of AP's active data transmission; *BUSY_time*: duration when channel is busy with other transmissions; *IDLE_slot_time*: count of idle channel slots. We poll these microsecond-precision counters every 1 ms and calculate the observed MAR by tracking changes in counter values. This provides accurate measurement of N_{tx} and N_{idle} as defined in §4.2.1. For CW control, we implement the complete HIMD algorithm with: observation interval of 300 slots when calculating MAR (justified in §J) and standard BE queue parameters ($CW_{min} = 15$, $CW_{max} = 1023$). The implementation consists of approximately 500 lines of C code, focusing on counter monitoring and CW adjustment logic.

6 Evaluation

We evaluate BLADE's performance through extensive experiments on commercial Wi-Fi APs and ns3 simulations. The assessment covers diverse network conditions, from saturated links to realistic traffic, culminating in real-world tests with cloud gaming applications. We benchmark BLADE against the standard IEEE 802.11 contention control and other relevant algorithms to validate its effectiveness.

6.1 Trace-driven Simulation

We use ns3 for our experimental environment because it accurately simulates the CSMA/CA behavior of Wi-Fi networks [33, 34] and allows easy modification of the contention control policy. For PHY transmission rate selection, we use Minstrel [35], the default rate adaptation algorithm in both ns3 and the `mac80211` module of Linux kernel.

Baselines. We evaluate BLADE against the following contention window control mechanisms:

- **BLADE SC**: BLADE with only stable-state control logic (*i.e.,* HIMD) to demonstrate the effectiveness of the fast recovery policy for collisions;
- **IEEE**: The default policy in the IEEE 802.11 standard, as explained in §3.2, using the BE (Best Effort) AC queue ($CW_{min} = 15, CW_{max} = 1023$);
- **IdleSense** [28]: It observes the mean number of idle slots between transmission attempts to control the contention window. We provide the transmitter number N to it as it requires such information to operate;
- **DDA** [29]: It controls the contention window to match the backoff delay threshold Δ imposed by applications. We Δ to be 5 ms (99th percentile value in Fig. 29).

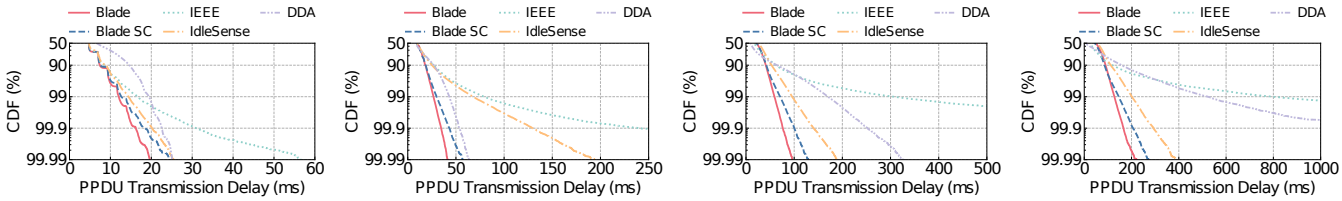
6.1.1 Saturated Link

Experimental Setup. To evaluate the performance of BLADE under intensive contention, we deploy N AP-STA pairs ($N = 2, 4, 8, 16$), each transmitting traffic from AP to STA using `iperf` to saturate the link. The evaluation utilizes the 802.11ax standard (Wi-Fi 6) operating in the 5 GHz band with a 40 MHz bandwidth. All transmitters share the same channel and can hear each other with equal signal strength.

AP Transmission Latency. To demonstrate BLADE's effectiveness in reducing tail latency for Wi-Fi last hop, we first evaluate the PPDU transmission latency (*i.e.,* frame exchange sequence duration in Fig. 2) to show how long a PPDU blocks the AP sending queue. As shown in Fig. 10, as the number of competing flows increases from 2 to 16, the median latency remains similar across all methods. However, the tail latency increases rapidly for the IEEE 802.11 standard contention control policy, exceeding 300 ms at the 99th percentile with 8 competing flows. In contrast, BLADE achieves the lowest tail latency among all methods, limiting the 99.99th percentile latency to 200 ms even with 16 competing flows. Notably, under 16 competing flows and the standard contention control policy, we observe frequent AP-STA disconnections due to Beacon frames experiencing excessively long contention intervals before transmission. This indicates that standard contention control policy fails to operate effectively under such high contention levels. Additionally, BLADE without the fast recovery policy shows a slight increase in tail latency, highlighting the effectiveness of BLADE's fast recovery mechanism.

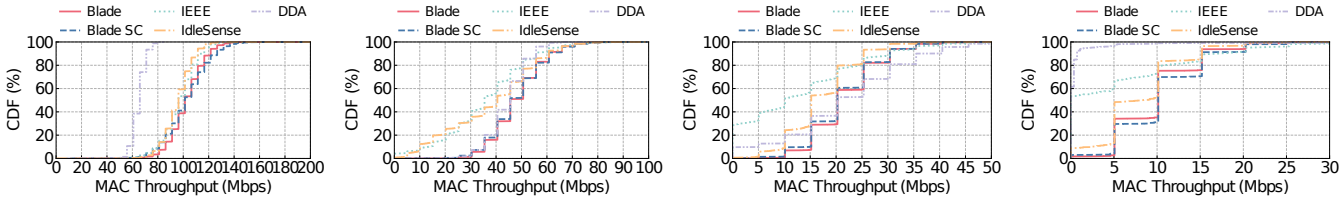
Retransmission Rate. To show BLADE's effectiveness in avoiding collisions and improving transmission efficiency, we plot the distribution of PPDU retransmission counts under 8 competing flows ($N = 8$) in Fig. 12. Thanks to the stable state control policy, BLADE adapts the contention windows of all transmitters to the channel contention level, achieving a low retransmission rate. Specifically, only 10% of PPDUs are retransmitted once, and 1% are retransmitted twice. In contrast, under intensive contention, the standard contention control policy results in 34% of PPDUs being retransmitted at least once, with 4% retransmitted more than twice.

MAC Throughput. We calculate the MAC throughput in 100 ms intervals and show the distribution in Fig. 11. Due to its lower PPDU retransmission rate and transmission latency,



(a) $N = 2$. (b) $N = 4$. (c) $N = 8$. (d) $N = 16$.

Figure 10: PPDU transmission delay distribution under N competing flows.



(a) $N = 2$. (b) $N = 4$. (c) $N = 8$. (d) $N = 16$.

Figure 11: Distribution of MAC throughput within 100 ms interval under N competing flows.

BLADE achieves higher median throughput compared to the standard policy as the number of competing flows increases. This result demonstrates that BLADE improves MAC layer transmission efficiency for Wi-Fi APs. Furthermore, BLADE results in a steadier and more converged throughput distribution. In contrast to the standard policy, BLADE prevents transient starvation, where the MAC throughput within 100 ms drops to zero, demonstrating that BLADE achieves fairer bandwidth allocation for all transmitters at the micro level.

Convergence & Fairness. To demonstrate the convergence of BLADE, we deploy five AP-STA pairs ($N = 5$) and sequentially start and stop their transmissions over a 5-minute period. As shown in Fig. 12a, with the arrival and departure of competing flows, the contention windows of all transmitters adapt dynamically to the contention level and converge within 1 second. Consequently, BLADE quickly achieves a fair bandwidth share among all transmitters, as illustrated in Fig. 12b.

6.1.2 Real-world Traffic

Experimental Setup. To evaluate BLADE's performance under real-world network traffic, we follow the simulation guidelines outlined in the IEEE standard [36] and simulate a three-floor apartment in ns3, as shown in Fig. 14. Each floor has eight rooms, each with one Wi-Fi AP (central-placed) and ten randomly distributed STAs forming a BSS. In every BSS, the AP sends two cloud gaming flows to two STAs, and the other STAs run real-world traffic trace(video streaming, web browsing, file transfer, etc.). We utilize four channels (*i.e.,* channel numbers 42, 58, 106, and 122) in the 5 GHz band with an 80 MHz bandwidth, ensuring that BSSes in adjacent rooms operate on different channels.

Traces. We use real-world open-source traces collected from routers [37] and base stations [38], covering traffic patterns used in our simulation. These traces include timestamps and packet sizes for packet arrivals in both downlink and uplink, representing traffic patterns at wireless last hops. For cloud gaming, we additionally access our cloud gaming platform and collect traffic traces directly from the Wi-Fi router.

Performance. Following the calculation in §6.1.1, we plot the PPDU transmission latency and MAC throughput for cloud gaming flows in Fig. 15 and Fig. 16. With contention from real-world competing network traffic, BLADE constrains the 99.9th and 99.99th percentile latency to 75 ms and 120 ms, respectively.In contrast, other methods inflate the tail latency to over 300 ms at the 99.99th percentile, while the standard control policy exceeds 500 ms. As a result, BLADE achieves only a 5% starvation rate (*i.e.,* MAC throughput within 100 ms drops to zero) across all methods, compared to the 25% starvation rate observed with the standard control policy. Notably, DDA and IdleSense perform worse than in the saturated link scenario because they assume *i.i.d.* traffic patterns from all competing flows, which is not true in real-world traffic.

6.2 Microbenchmarks

6.2.1 Influence of Target MAR

We evaluate the impact of the target MAR on the performance of BLADE. We repeat the experiment in §6.1.1 with $N = 4$ and MAR_{tar} varying from 0.05 to $MAR_{max} = 0.35$. As shown in Fig. 17, when MAR_{tar} deviates from the default value of 0.1 within ± 0.05, the performance of BLADE remains relatively stable, with a ± 5 ms tail PPDU transmission delay and a ± 2.5 Mbps median MAC throughput deviation. However, as MAR_{tar} approaches MAR_{max}, the tail latency increases rapidly, reaching 150% of the default value. These results align with our analysis in §F, which shows that $MAR_{tar} = 0.1$ is an appropriate and robust default value.

6.2.2 Parameter Sensitivity

BLADE is robust to parameter choices; varying M_{inc}, M_{dec}, A_{inc}, and A_{fail} yields negligible changes in throughput and

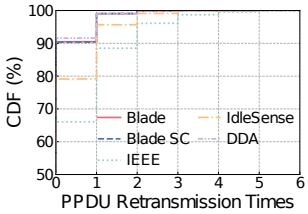
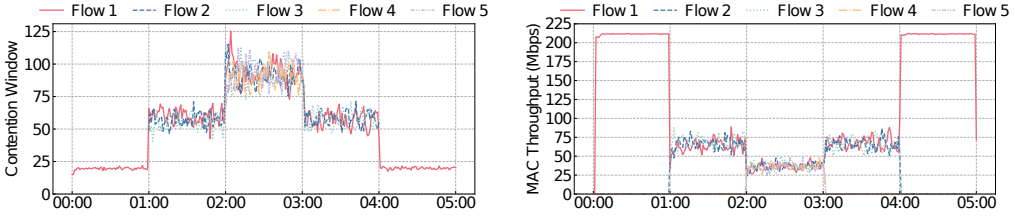


Figure 12: Retransmission times for each PPDU under 8 competing flows.



(a) Contention window value. (b) MAC throughput.

Figure 13: Convergence of BLADE with five competing flows.

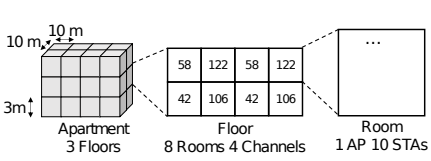


Figure 14: Simulation topology of an apartment.

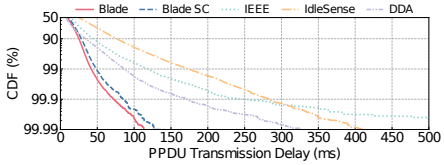


Figure 15: Cloud gaming flow PPDU transmission delay distribution.

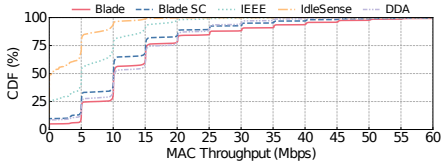
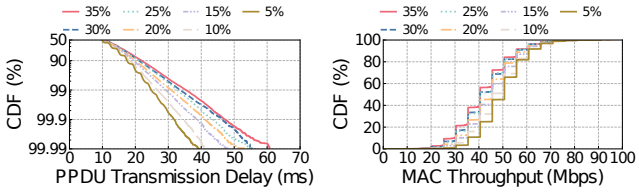


Figure 16: Cloud gaming flow MAC throughput within 100 ms interval.



(a) PPDU transmission delay. (b) MAC throughput.

Figure 17: Performance of BLADE under different target utilization rate MAR_{tar}.

Table 3: Mobile gaming packet latency distribution (%)

RTT (ms)	0 Competing Flow		1 Competing Flow		2 Competing Flows		3 Competing Flows	
	IEEE	Blade	IEEE	Blade	IEEE	Blade	IEEE	Blade
[0, 10)	99.7	99.8	12.4	88.6	2.1	85.9	2.3	84.1
[10, 20)	0.3	0.2	32.1	11.2	30.6	13.8	22.7	15.7
[20, 30)	0.0	0.0	28.3	0.2	29.5	0.1	27.8	0.1
[30, 40)	0.0	0.0	18.1	0.0	19.0	0.2	22.7	0.1
[40, 50)	0.0	0.0	5.3	0.0	10.1	0.0	11.3	0.0
[50,100)	0.0	0.0	3.8	0.0	8.7	0.0	13.2	0.0

PPDU TX-delay percentiles (details in §C.1).

6.3 Real-World Experiments

Experimental Setup. To evaluate the performance of BLADE in real world, we conduct experiments using commercial Wi-Fi APs with BLADE implemented. We deploy 4 AP-STA pairs. The evaluation utilizes the 802.11ax standard (Wi-Fi 6) operating in the 5 GHz band with 40 MHz bandwidth. All transmitters share the same channel and can hear each other.

6.3.1 Saturated Links

We first saturate the wireless link with 4 AP each transmitting an `iperf` flow to the STA. As shown in Fig. 18, BLADE consistently achieves lower tail PPDU transmission delay compared to the IEEE standard, with more than 4× reduction. Therefore, as shown in Fig. 19, BLADE achieves more stable and higher MAC bandwidth utilization than the IEEE standard, demonstrating better adaptation to real-world wireless channel dynamics and reducing inefficiencies caused by excessive contention window growth.

6.3.2 Cloud Gaming

We replay our cloud gaming session while injecting 0–3 contending `iperf` flows. Fig. 20 shows that BLADE keeps the 99th-percentile end-to-end frame delay below 100 ms under heavy contention (vs. $> 200\,ms$ for IEEE), cutting stall rate by $> 90\%$. This directly translates to smoother interactive gameplay, aligning with the motivation in §3.1.

6.3.3 Mobile Game Traffic

we compares the RTT distribution of a mobile game under different numbers of competing flows using the IEEE 802.11 standard and BLADE. all flows adopt the same CW adjustment algorithm during the experiment. As shown in Tab. 3,without competing flows, both strategies achieve ultra-low latency. However, IEEE 802.11 performance degrades sharply with more competing flows, with far fewer low-latency packets and higher RTTs. In contrast, BLADE maintains over 84% of packets within [0,10) ms even under three competing flows, while IEEE 802.11 drops to only 2.3%. This demonstrates BLADE's effectiveness in mitigating contention and ensuring low-latency performance for mobile gaming applications.

6.3.4 File Downloading

Tab. 4 presents the speed distribution under different contention levels while downloading a large file, comparing IEEE 802.11 with BLADE. Without competing flows, both schemes maintain speeds above 50 Mbps. However, IEEE degrades with 40% of traffic below 50 Mbps under one competing flow (BLADE keeps 94% at 10–50 Mbps). Under heavy contention, 50% of IEEE traffic drops below 10 Mbps,while 67% BLADE traffic exceeds 20 Mbps. These results proving BLADE mitigates contention-induced degradation for more stable throughput.

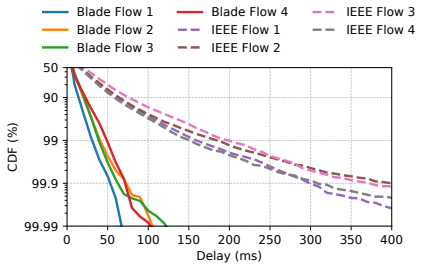


Figure 18: Distribution of transmission delay for four `iperf` flows.

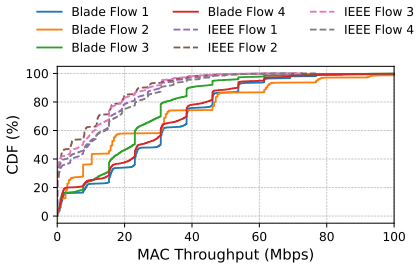


Figure 19: Distribution of MAC throughput for four `iperf` flows.

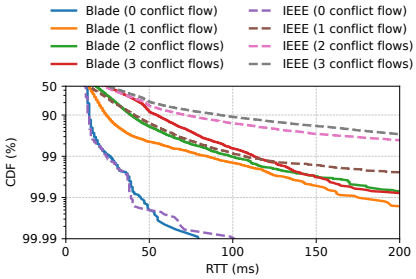


Figure 20: End-to-end frame delay under varying number of `iperf` flows.

Table 4: Download bandwidth distribution under different contention levels (%)

Bandwidth	0 Flow		1 Flow		2 Flows		3 Flows	
(Mbps)	IEEE	Blade	IEEE	Blade	IEEE	Blade	IEEE	Blade
0–5	0	0	1	0	0	0	1	0
5–10	0	0	5	0	43	1	79	0
10–20	0	0	50	2	57	17	10	24
20–30	0	0	3	3	0	71	0	74
30–40	0	0	0	52	0	9	0	2
40+	100	100	41	43	0	2	0	0

Overall, these real-world experiments demonstrate that BLADE effectively optimizes CW adjustment, accommodating both throughput and latency-sensitive applications.

7 Discussion

Why not centralized scheduling? Explicit scheduling (e.g., TDMA-style airtime reservation) avoids contention, but requires tight coordination and a common administrative domain, which is unachievable for Wi-Fi routers purchased and controlled by end consumers who lack centralized management capabilities.In contrast, BLADE operates within CSMA/CA, only adjusting local contention windows (CW). Its fully distributed design enables incremental deployment in commodity APs without neighboring coordination.

Coexistence with IEEE 802.11 Contention Control. BLADE achieves fair convergence when universally deployed, but may inadvertently cede transmission opportunities to IEEE 802.11-compliant devices (which typically retain small contention windows). As shown in §G,configuring BLADE with a higher MAR_{tar} enhances its competitiveness with legacy devices. Notably, BLADE supports incremental deployment: even on partial adoption, it suppresses contention-driven packet-delivery droughts for its own traffic, while full deployment maximizes fairness and latency performance.

Hidden Terminal. Since BLADE relies on the consensus signal from the same channel, it may be affected by the Hidden Terminal Problem [39], where transmitters perceive different transmission opportunity utilization rates. In this setting, MAR should be interpreted as a "local" contention signal within a carrier-sensing domain rather than a globally consistent metric. RTS/CTS is widely used to mitigate this issue. Since a CTS is followed by a PPDU transmission from a hidden terminal, upon receiving CTS, BLADE can infer that two transmission opportunities have been utilized when calculating MAR. We show in §H that BLADE maintains low PPDU transmission delay for all transmitters in the presence of hidden terminals.

8 Related Work

Real-Time Streaming in Wireless Networks. Many prior studies have discussed the latency bottleneck of wireless networks in real-time streaming. They propose various methods to reduce the tail frame delivery latency, including congestion control algorithms [8], multipath transmission [9], and novel loss recovery schemes at both transport layer [11] and application layer [40]. These studies regard wireless fluctuations as inherent to the link and rely on *indirect* solutions from higher layers to alleviate its impact. Orthogonal to all these methods, BLADE takes a *direct* approach on the link layer to mitigate the long tail latency induced by Wi-Fi last hop and can be jointly deployed with them.

Wi-Fi Performance Enhancement. Prior work improves Wi-Fi via rate adaptation, contention control, channel selection, and AQM [28, 29, 35, 41–43]. Few focus on latency-sensitive flows [10, 44]. BLADE targets contention-driven tail latency with no traffic-pattern assumptions.

9 Conclusion

In this paper, we reveal the fundamental limitation of the contention window adjustment mechanism in IEEE 802.11 standard and identify it to be the root cause of long tail video frame delivery latency of next-generation real-time communication applications (NGRTC) in Wi-Fi networks. We present BLADE, a novel contention window control algorithm. Compared to the standard, BLADE can significantly reduce Wi-Fi transmission latency and improve MAC throughput. We believe BLADE to be an important building block towards the rapid development of NGRTC.

Acknowledgments

We thank our shepherd, Robert Ricci, and the anonymous reviewers for their valuable comments. We would like to express our sincere gratitude to our colleagues at Tencent START, including Weiting Xiao, Zhenxing Wen, Jianjun Xiao, Nian Wen and Jiafeng Chen, for their invaluable technical support, insightful discussions.

References

[1] Samsung Gaming Hub. https://www.samsung.com/us/televisions-home-theater/tvs/gaming-hub/. (Accessed on 01/02/2024).

[2] Google Cloud Gaming. https://cloud.google.com/solutions/games. (Accessed on 01/02/2024).

[3] Google map live view support. https://support.google.com/maps/answer/9332056?hl=en&co=GENIE.Platform%3DiOS, 2023.

[4] Youtube vr - home. https://vr.youtube.com/, 2023.

[5] Cloud gaming market: Global industry trends, share, size, growth, opportunity and forecast 2023-2028. Market report 5732901, IMARC Group, 2023.

[6] Xiaokun Xu and Mark Claypool. Measurement of cloud-based game streaming system response to competing TCP cubic or TCP BBR flows. In *ACM IMC*, 2022.

[7] Simone Mangiante, Guenter Klas, Amit Navon, Zhuang GuanHua, Ju Ran, and Marco Dias Silva. VR is on the Edge: How to Deliver 360° Videos in Mobile Networks. In *ACM VR/AR Network*, 2017.

[8] Shibo Wang, Shusen Yang, Xiao Kong, Chenglei Wu, Longwei Jiang, Chenren Xu, Cong Zhao, Xuesong Yang, Jianjun Xiao, Xin Liu, Changxi Zheng, Jing Wang, and Honghao Liu. Pudica: Toward Near-Zero queuing delay in congestion control for cloud gaming. In *USENIX NSDI*, 2024.

[9] Yuhan Zhou, Tingfeng Wang, Liying Wang, Nian Wen, Rui Han, Jing Wang, Chenglei Wu, Jiafeng Chen, Longwei Jiang, Shibo Wang, Honghao Liu, and Chenren Xu. AUGUR: Practical mobile multipath transport service for low tail latency in Real-Time streaming. In *USENIX NSDI*, 2024.

[10] Zili Meng, Yaning Guo, Chen Sun, Bo Wang, Justine Sherry, Hongqiang Harry Liu, and Mingwei Xu. Achieving Consistent Low Latency for Wireless Real-Time Communications with the Shortest Control Loop. In *ACM SIGCOMM*, 2022.

[11] Zili Meng, Xiao Kong, Jing Chen, Bo Wang, Mingwei Xu, Rui Han, Honghao Liu, Venkat Arun, Hongxin Hu, and Xue Wei. Hairpin: Rethinking Packet Loss Recovery in Edge-based Interactive Video Streaming. In *USENIX NSDI*, 2024.

[12] Zili Meng, Tingfeng Wang, Yixin Shen, Bo Wang, Mingwei Xu, Rui Han, Honghao Liu, Venkat Arun, Hongxin Hu, and Xue Wei. Enabling High Quality Real-Time Communications with Adaptive Frame-Rate. In *USENIX NSDI*, 2023.

[13] Jiangkai Wu, Yu Guan, Qi Mao, Yong Cui, Zongming Guo, and Xinggong Zhang. ZGaming: Zero-Latency 3D Cloud Gaming by Image Prediction. In *ACM SIGCOMM* , 2023.

[14] 3GPP TR 38.211 (Release 16). 5G; NR; Physical channels and modulation. https://www.etsi.org/deliver/etsi_ts/138200_138299/138211/16.02.00_60/ts_138211v160200p.pdf, 2020.

[15] 3GPP TR 38.213 (Release 16). 5G; NR; Physical Layer Procedures for Control. https://www.etsi.org/deliver/etsi_ts/138200_138299/138213/16.02.00_60/ts_138213v160200p.pdf, 2020.

[16] Cloudflare Radar 2025 Review. https://blog.cloudflare.com/radar-2025-year-in-review/.

[17] Rachel Albert, Anjul Patney, David Luebke, and Joohwan Kim. Latency requirements for foveated rendering in virtual reality. *ACM Trans. Appl. Percept.*, 14(4), September 2017.

[18] Richard Yao, Tom Heath, Aaron Davies, Tom Forsyth, Nate Mitchell, and Perry Hoberman. *Oculus VR Best Practices Guide*. Oculus VR, Inc., April 2014. Version 0.008 (April 30, 2014).

[19] Sara Vlahovic, Mirko Suznjevic, and Lea Skorin-Kapov. The Impact of Network Latency on Gaming QoE for an FPS VR Game. In *IEEE QoMEX*, 2019.

[20] Mohammed S. Elbamby, Cristina Perfecto, Mehdi Bennis, and Klaus Doppler. Toward Low-Latency and Ultra-Reliable Virtual Reality. *IEEE Network*, 32(2):78–84, 2018.

[21] Eugene Korneev, Mikhail Liubogoshchev, Dmitry Bankov, and Evgeny Khorov. How to model cloud vr: An empirical study of features that matter. *IEEE Open Journal of the Communications Society*, 5:4155–4170, 2024.

[22] IEEE 802.11ax Standard . https://standards.ieee.org/ieee/802.11ax/7180/. (Accessed on 22/10/2024).

[23] Kun Tan, Ji Fang, Yuanyang Zhang, Shouyuan Chen, Lixin Shi, Jiansong Zhang, and Yongguang Zhang. Fine-grained channel access in wireless lan. In *Proceedings of the ACM SIGCOMM 2010 Conference*, pages 147–158. ACM, 2010.

[24] Souvik Sen, Romit Roy Choudhury, and Srihari Nelakuditi. No time to countdown: Migrating backoff to the frequency domain. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 241–252. ACM, 2011.

[25] Souvik Sen, Romit Roy Choudhury, and Srihari Nelakuditi. Csma/cn: Carrier sense multiple access with collision notification. In *Proceedings of the 16th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 25–36. ACM, 2010.

[26] Vivek Shrivastava, Nabeel Ahmed, Shravan Rayanchu, Suman Banerjee, Srinivasan Keshav, Konstantina Papagiannaki, and Arunesh Mishra. Centaur: Realizing the full potential of centralized wlans through a hybrid data path. In *Proceedings of the 15th Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 297–308. ACM, 2009.

[27] Martin Heusse, Franck Rousseau, Romaric Guillier, and Andrzej Duda. Idle sense: An optimal access method for high throughput and fairness in rate diverse wireless lans. In *Proceedings of the ACM SIGCOMM 2005 Conference*, pages 121–132. ACM, 2005.

[28] Martin Heusse, Franck Rousseau, Romaric Guillier, and Andrzej Duda. Idle sense: an optimal access method for high throughput and fairness in rate diverse wireless lans. In *Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '05, 2005.

[29] Y. Yang and R. Kravets. Achieving delay guarantees in ad hoc networks through dynamic contention window adaptation. In *IEEE INFOCOM*, 2006.

[30] Xuejun Tian, Xiang Chen, Tetsuo Ideguchi, and Yuguang Fang. Improving throughput and fairness in wlans through dynamically optimizing backoff. *IEICE transactions on communications*, 88(11):4328–4338, 2005.

[31] Qin Yu, Yiqun Zhuang, and Lixiang Ma. Dynamic contention window adjustment scheme for improving throughput and fairness in ieee 802.11 wireless lans. In *IEEE Global Communications Conference (GLOBECOM)*, 2012.

[32] Qiang Ni, I. Aad, C. Barakat, and T. Turletti. Modeling and analysis of slow cw decrease ieee 802.11 wlan. In *14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications, 2003.*, 2003.

[33] Nicola Baldo, Manuel Requena-Esteso, José Núñez-Martínez, Marc Portolès-Comeras, Jaume Nin-Guerrero, Paolo Dini, and Josep Mangues-Bafalluy. Validation of the ieee 802.11 mac model in the ns3 simulator using the extreme testbed. In *Proceedings of the 3rd International ICST Conference on Simulation Tools and Techniques*, pages 1–9, 2010.

[34] NS3 Wi-Fi Validation against Bianchi Model. https://www.nsnam.org/docs/release/3.41/models/html/wifi-testing.html#bianchi-validation. (Accessed on 12/23/2024).

[35] Andrew Mcgregor and Derek Smithies. Rate adaptation for 802.11 wireless networks: Minstrel. *Submitted to ACM SIGCOMM*, 2010.

[36] "TGax Simulation Scenarios", IEEE 802.11-14/0980r16. https://mentor.ieee.org/802.11/dcn/14/11-14-0980-16-00ax-simulation-scenarios.docx. (Accessed on 12/10/2024).

[37] VPN/Non-VPN Network Application Traffic Dataset (VNAT). https://www.ll.mit.edu/r-d/datasets/vpnnonvpn-network-application-traffic-dataset-vnat. (Accessed on 01/08/2025).

[38] 5G Traffic Datasets. https://www.kaggle.com/datasets/kimdaegyeom/5g-traffic-datasets. (Accessed on 01/08/2025).

[39] Wi-fi hidden terminal problem. https://en.wikipedia.org/wiki/Hidden_node_problem, 2025.

[40] Yihua Cheng, Ziyi Zhang, Hanchen Li, Anton Arapin, Yue Zhang, Qizheng Zhang, Yuhan Liu, Kuntai Du, Xu Zhang, Francis Y. Yan, Amrita Mazumdar, Nick Feamster, and Junchen Jiang. GRACE: Loss-Resilient Real-Time Video through Neural Codecs. In *USENIX NSDI*, 2024.

[41] Mathieu Lacage, Mohammad Hossein Manshaei, and Thierry Turletti. IEEE 802.11 rate adaptation: a practical approach. In *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, 2004.

[42] S. Vasudevan, K. Papagiannaki, C. Diot, J. Kurose, and D. Towsley. Facilitating Access Point Selection in IEEE 802.11 Wireless Networks. In *ACM IMC*, 2005.

[43] Toke Høiland-Jørgensen, Michał Kazior, Dave Täht, Per Hurtig, and Anna Brunstrom. Ending the anomaly: achieving low latency and airtime fairness in WiFi. In *USENIX ATC*, 2017.

[44] Changhua Pei, Youjian Zhao, Yunxin Liu, Kun Tan, Jiansong Zhang, Yuan Meng, and Dan Pei. Latency-based WiFi congestion control in the air for dense WiFi networks. In *IEEE IWQoS*, 2017.

[45] IEEE 802.11e Standard . https://standards.ieee.org/ieee/802.11e/3131/. (Accessed on 22/10/2024).

[46] Giuseppe Bianchi. Performance analysis of the ieee 802.11 distributed coordination function. *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, 18:535, 2000.

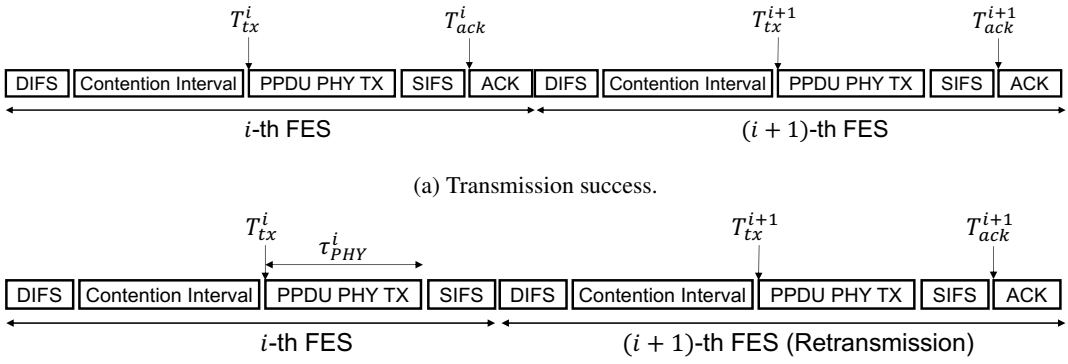
A PPDU Contention Interval Calculation

Since the random backoff procedure is handled by WNIC firmware, it is not easy to directly acquire the contention interval for each PPDU transmitted in Wi-Fi. Therefore, we adopt a passive approach: we deploy an AP-STA pair and use `iperf` to keep the AP WNIC busy. We place a Wi-Fi sniffer closely to the AP to capture all traffic (including PPDUs and ACKs) related to the AP. As shown in Fig. 21a, because the WNIC keeps busy, the frame exchange sequence (FES) of the i-th PPDU is immediately followed by the FES of the $(i+1)$-th PPDU. From the sniffed trace, we can acquire the precise timestamp of PHY transmission event T_{tx}^{i} and ACK event T_{ack}^{i} of the i-th PPDU. Since the DIFS, SIFS and ACK are standard intervals with fixed values, we can calculate the contention interval of the $(i+1)$-th PPDU as $T_{tx}^{i+1} - T_{ack}^{i} - ACK - DIFS$. The PHY transmission time of the i-th PPDU can be calculated as $T_{ack}^{i} - T_{tx}^{i} - SIFS$.

Upon transmission failures, no ACK frame is sniffed. As illustrated in Fig. 21b, in this case, we can calculate the PHY transmission time of the i-th PPDU τ_{PHY}^{i} as we can acquire the PPDU size and transmission rate from the sniffed trace. Therefore, the contention interval of the $(i+1)$-th PPDU can be calculated as $T_{tx}^{i+1} - T_{tx}^{i} - \tau_{PHY}^{i} - SIFS - DIFS$.

B Limitation of Priority-Based IEEE 802.11 Contention Control

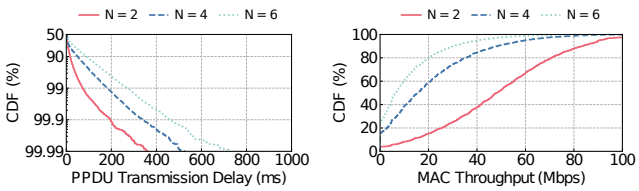
IEEE 802.11e standard [45] has proposed Enhanced Distributed Channel Access (EDCA), defining several Access Categories (ACs) queues with different CW_{min} and CW_{max} values to accommodate various QoS requirements. Specifically, there are BK (Background) queue with $CW_{min} = 7, CW_{max} = 1023$ BE (Best Effort) queue with $CW_{min} = 15, CW_{max} = 1023$, VI (Video) queue with $CW_{min} = 7, CW_{max} = 15$, and VO (Voice) queue with $CW_{min} = 1, CW_{max} = 3$. While the BE queue is the default, adopting AC queues with higher priority can lower the CW value and occupy more transmission chances. However, with the rapid development of RTC applications, it is common to have multiple RTC sessions in the same wireless environment nowadays. When multiple flows with high priority contend for transmission chances in the same channel, the contention level can be severely intensified, leading to more collisions. We demonstrate this by repeating the experiment in §6.1.1, with N ($N = 2, 4, 6$) `iperf` flows saturating the link with the VI queue. As shown in Fig. 22, with competing flows from VI queues, the PPDU transmission delay significantly increases even with $N = 2$, compared to the BE queue (the 99.99th percentile delay is 56 ms for BE queue, as shown in Fig. 10a). Therefore, the MAC throughput within 100 ms interval exhibits a more unsteady pattern, with 19% starvation rate when $N = 4$ (the starvation rate for BE queue when $N = 4$ is 4%, as shown in Fig. 11b).



(a) Transmission success.

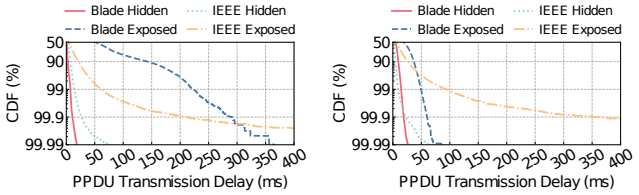
(b) Transmission failure.

Figure 21: Illustration of PPDU contention interval calculation.



(a) PPDU transmission delay. (b) MAC throughput.

Figure 22: Performance of VI AC queue with N competing flows.



(a) RTS/CTS disabled. (b) RTS/CTS enabled.

Figure 23: Influence of hidden terminal on BLADE with RTS/CTS disabled/enabled.

C Additional Evaluation Results

C.1 Parameter Sensitivity

We evaluate the parameter sensitivity of BLADE by repeating the experiment in §6.1.1 with $N = 4$ and different parameter values. As shown in Tab. 5, changes in parameter values lead to negligible performance shifts compared to the default configuration. Therefore, BLADE is robust and not sensitive to its parameters.

D Detailed Anatomy of Packet Delivery Droughts

This appendix expands the mechanism summary in §3.2. We provide (i) ns-3 simulation evidence that collisions increase retransmissions under multi-AP contention, (ii) controlled AP experiments validating the same effect in practice, and (iii) a packet-level example and delay decomposition illustrating how countdown freezes amplify contention intervals.

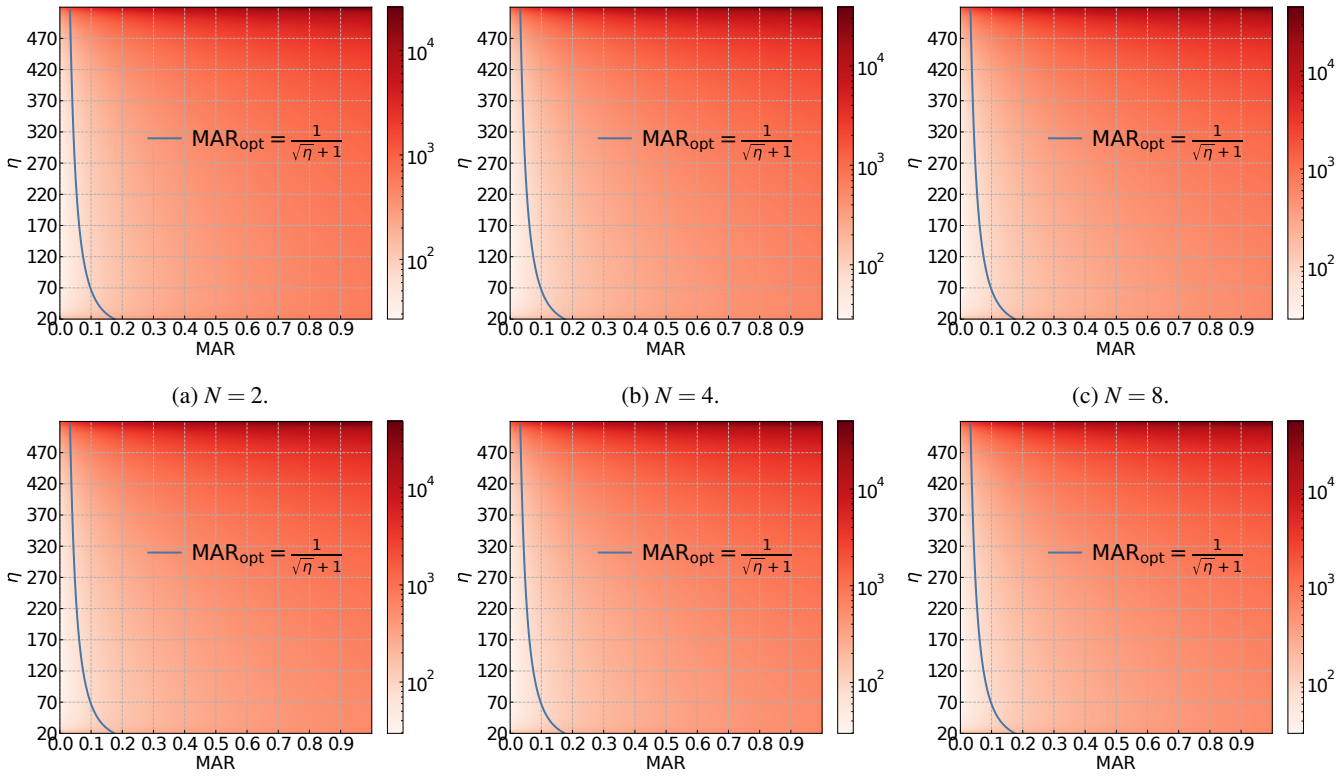
Variant	Avg. MAC Throughput (Mbps)	50/95/99/99.9/99.99th PPDU TX Delay (ms)
BLADE Default	48.5	9.8/21.0/26.7/34.6/42.1
$M_{inc} = 250$	48.1	9.8/21.2/27.3/35.2/43.1
$M_{inc} = 125$	48.6	9.7/21.0/26.6/34.3/42.4
$M_{dec} = 0.85$	48.5	9.7/21.4/27.7/36.5/44.9
$M_{dec} = 0.75$	48.1	9.7/21.9/28.6/37.8/46.4
$A_{inc} = 10$	48.2	9.8/21.2/26.9/35.0/42.1
$A_{inc} = 30$	48.9	9.6/21.7/28.2/37.1/46.1
$A_{fail} = 10$	48.6	9.7/21.2/27.1/35.3/42.5
$A_{fail} = 20$	48.7	9.7/21.7/28.1/37.7/44.4

Table 5: Parameter Sensitivity of BLADE.

To understand why packets fail to be delivered within measurement intervals, we conducted systematic NS3 simulations, which accurately model Wi-Fi's CSMA/CA behavior [33, 34]. We deploy N Wi-Fi APs contending in the same channel, each transmitting `iperf` flows to saturate the link. Our analysis reveals two key factors that lead to packet delivery droughts:

First, packet retransmissions significantly extend the total delivery time, as each failed attempt requires additional transmission attempts. Fig. 26 demonstrates how channel contention directly impacts retransmission frequency: with 2 competing devices ($N = 2$), almost all packets are delivered successfully on first attempt. However, as contention increases, packets require more retransmission attempts. When eight devices compete ($N = 8$), 34% of packets need at least one retransmission, with some requiring up to 6 retries. This means a single packet can require multiple transmission attempts before successful delivery, substantially extending its delivery time.

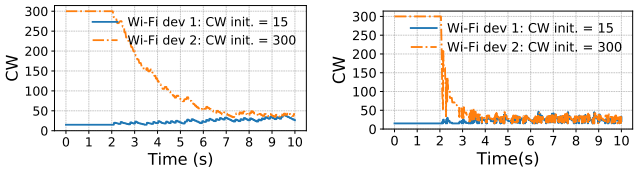
Second, and more critically, each retransmission triggers a vicious cycle due to Wi-Fi's exponential backoff mechanism. When a transmission fails, the device doubles its contention window, effectively reducing its ability to compete for channel access compared to devices with smaller windows. Fig. 27 demonstrates this effect by tracking contention intervals across successive retransmission attempts (with $N = 6$). While the first transmission attempt experiences relatively



(a) $N = 2$. (b) $N = 4$. (c) $N = 8$.

(d) $N = 16$. (e) $N = 32$. (f) $N = 64$.

Figure 24: $\mathcal{L}(MAR)$ dynamics with respect to different values of MAR and η under different transmitter numbers N. The blue line illustrates the optimal MAR value for each η.



(a) Traditional AIMD. (b) BLADE's HIMD

Figure 25: The comparison of convergence speed of traditional AMID and HIMD of BLADE.

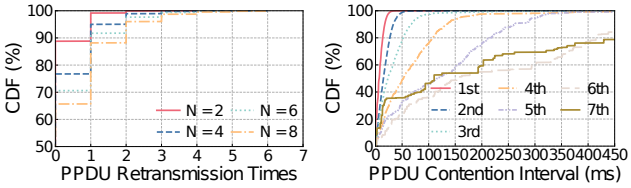


Figure 26: PPDU retransmission times with N competing flows.

Figure 27: PPDU backoff time at the n-th transmission with $N = 6$.

short contention intervals, each subsequent attempt faces progressively longer delays due to the enlarged contention window. By the sixth retransmission, over 60% of PPDUs experience contention intervals exceeding 200ms. As a result, the transmission delay for each PPDU (*i.e.,* frame exchange sequence duration in Fig. 2) significantly increases with competing flow number N, as shown in Fig. 28. This combination of frequent retransmissions and extended contention intervals explains the packet delivery droughts we observed in production networks.

D.0.1 Validating Simulation Results with Wi-Fi AP

To complement our simulation findings and validate them in real-world conditions, we conducted controlled experiments using commercial Wi-Fi APs. While our simulations provide insights into the fundamental relationship between contention and packet delivery, real-world factors such as channel dynamics and interference could affect these behaviors. We set up a testbed using Xiaomi AX3600 Wi-Fi APs in a typical office environment, where they experience natural channel contention from existing network traffic. Through these experiments, we aim to verify whether the packet delivery patterns and retransmission behaviors observed in simulations manifest similarly in practice.

In our experiments, we established a saturated link between the AP and a client device (STA) using `iperf`, allowing us to observe the system under consistent load. By analyzing air-sniffed traces, we calculated precise PHY transmission delay and contention interval for each PPDU (detailed methodology in §A). This controlled setting enabled us to dissect the transmission process and identify the key factors leading to packet delivery drought.

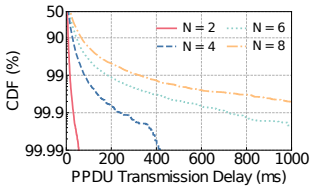


Figure 28: PPDU transmission delay with N competing flows.

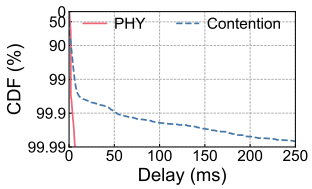


Figure 29: Contention interval and PHY latency distribution for each Wi-Fi PPDU.

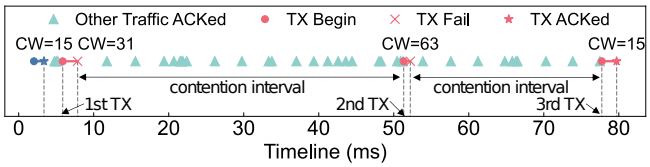


Figure 30: Lifetime of a single PPDU (in red). Green triangles illustrate the competing traffic from other devices.

An Example of Extended Packet Delivery Time. Fig. 30 shows how a single packet's delivery time can extend to 75.9ms, orders of magnitude longer than expected. This significant extension stems from two factors. First, after each collision, the PPDU requires a retransmission attempt, and our example packet needs multiple retransmissions. Second, during each retransmission attempt, the contention interval becomes severely extended. While the doubled contention window only increases from CW=15 (max of 135μs) to CW=31 (max of 279μs), the actual contention intervals stretch to 43.5ms and 25.5ms because other devices (green triangles) repeatedly gain channel access during the countdown process. Each time another device transmits, our PPDU must freeze its countdown. Through this combination of multiple retransmission attempts and extended contention intervals, what should be a quick packet delivery becomes a 75.9ms process.

Statistical Analysis of Contention Intervals. To quantify this phenomenon, we analyzed the distribution of both PHY transmission times and contention intervals across all PPDUs (Fig. 29). PHY transmission time—the actual time spent transmitting data over the air as shown in Fig. 2—remains predictably brief ($<$ 5ms at 99.99th percentile) due to fixed Wi-Fi hardware constraints. In stark contrast, contention intervals, representing time spent competing for channel access, exhibit alarming variability. While their median stays below 1ms, the tail extends dramatically, exceeding 200ms at the 99.99th percentile. This means Wi-Fi devices spend orders of magnitude more time competing for transmission opportunities than actually transmitting data.

Takeaway. Our real-world measurements validate that packet delivery drought stems from both retransmission attempts and extended contention intervals, with devices spending up to 200ms competing for channel access compared to just 5ms for actual data transmission.

E MAR-Driven CW Control Algorithm

The detailed pseudo code of the MAR-driven CW control algorithm is shown in Alg. 1.

F Target MAR Analysis

The target microscopic access rate MAR_{tar} plays an essential role in BLADE's stable-state control policy. Here, we analyze the impact of MAR_{tar} and discuss the criteria for selecting its optimal value.

F.1 Inverse Proportion

We first analyze the relationship between the CW value and the microscopic access rate MAR. For a transmitter i with contention window CW_i, the probability τ_i of attempting a transmission at any given transmission chance (highlighted in red in Fig. 9) is the probability that its random backoff timer reaches zero at that moment:

$$\tau_i = \frac{CW_i}{\sum_{k=1}^{CW_i} k} = \frac{2}{CW_i + 1} \tag{7}$$

In the stable state, the contention windows of all transmitters converge to the same value CW (i.e., $\tau_i = \tau = \frac{2}{CW+1}$). With N transmitters, the probabilities for a transmission chance being idle, occupied by a successful transmission, or resulting in a collision are as follows:

$$P_i = (1-\tau)^N,\ P_s = N\tau(1-\tau)^{N-1},\ P_c = 1 - P_i - P_s \tag{8}$$

The microscopic access rate MAR represents the probability that a transmission chance is used:

$$MAR = 1 - P_i = 1 - (1-\tau)^N \overset{\tau \ll 1}{\approx} N\tau = \frac{2N}{CW+1} \tag{9}$$

Since CW values are typically much larger than 1, we apply a first-order approximation in Eqn. 9. This shows that, in the stable state, the microscopic access rate MAR is inversely proportional to the converged contention window value CW.

F.2 Robustness

Here, we discuss the selection of the target MAR. In the stable state, the micro-level bandwidth fairness is achieved as all transmitters' CW values converge to the same. Therefore, our target MAR aims to maximize the overall transmission bandwidth. Suppose the average PPDU size is S, the average time of a successful transmission, collided transmission, and slot time is T_t, T_c, and T_s, respectively. Note that BLADE's design principle and control logic do not rely on these assumptions. Similar to the derivation in [28], the overall throughput can be presented as:

$$Thp = \frac{P_t S}{P_s T_t + P_c T_c + P_i T_s} = \frac{S}{T_t + \frac{(1-P_i-P_s)\eta + P_i}{P_s} \cdot T_s} \tag{10}$$

where $\eta = T_c/T_s$. To maximize Thp, combined with Eqn. 8 and Eqn. 9, we only have to minimize the cost function:

$$\mathcal{L}(MAR) = \frac{(1-P_i-P_s)\eta + P_i}{P_s} = \frac{N-MAR}{N} \cdot \frac{(\eta-1)MAR+1}{MAR(1-MAR)} \quad (11)$$

The optimal value MAR_{opt} is determined by N and η. However, since $MAR < 1$ and $N \geq 2$, MAR has a negligible effect on the first term $\frac{N-MAR}{N}$. As a result, MAR_{opt} is almost independent of the number of transmitters N. Next, consider the second term $\frac{(\eta-1)MAR+1}{MAR(1-MAR)}$. Taking the derivative of this term and setting it equal to zero yields:

$$MAR_{opt} = \frac{1}{\sqrt{\eta}+1} \quad (12)$$

Since η represents the number of time slots occupied by a collided transmission, its average value depends on the PHY transmission time of each PPDU, which in turn is determined by the PPDU size and PHY transmission rate. In the 802.11ax (Wi-Fi 6) standard, η can range from 20 to over 500. More importantly, although the optimal value of MAR is primarily determined by η, the cost function $\mathcal{L}(MAR)$ is relatively insensitive to changes in MAR. We illustrate this by plotting $\mathcal{L}(MAR)$ against different values of MAR and η in a heatmap (Fig. 24) with increasing N values. As MAR deviates from MAR_{opt}, the change in $\mathcal{L}(MAR)$ is minimal, and such pattern does not change with the increase of N value. Therefore, BLADE is robust to the selection of MAR_{tar}, as long as it remains within a "safe zone" (± 0.1) around MAR_{opt}. As a result, from Fig. 24, we set the default value of MAR_{tar} to be 0.1.

G Coexistence with IEEE 802.11 Standard Contention Control

Similar to the experimental setup in §6.1.1, we deploy four AP-STA pairs, with two pairs running BLADE and the other two pairs using the IEEE 802.11 standard contention control policy. The APs saturate the wireless link by sending `iperf` traffic to the STAs. As shown in Tab. 6, with MAR_{tar} increasing from 0.1 to 0.5, BLADE becomes more competitive to the standard control policy and gains more MAC throughput with lower PPDU transmission delay. Therefore, BLADE can be configured with higher MAR_{tar} values when coexisting with the standard policy, while still ensuring convergence and fairness when all APs implement BLADE.

H Influence of Hidden Terminal

Since BLADE relies on the consensus signal from the same channel, it may be affected by the Hidden Terminal Problem [39]. To demonstrate this impact, similar to the experimental setup in §6.1.1, but we deploy the AP-STA pairs in three rooms arranged in a row. Transmitters at both ends cannot hear each other, acting as hidden terminals, while transmitters in the middle can hear traffic from both ends, acting as exposed terminals. We show the PPDU transmission delay distribution for hidden and exposed terminals in Fig. 23. When RTS/CTS is disabled, both BLADE and the IEEE 802.11 standard contention control policy result in increased tail latency for exposed terminals, as they undergo more intensive contention. However, with RTS/CTS enabled, because BLADE counts CTS signals in opportunity utilization rate calculation, BLADE shows much smaller differences in delay distribution between exposed and hidden terminals. These results demonstrate that BLADE is compatible with the widely deployed RTS/CTS mechanism and is robust to the Hidden Terminal Problem when RTS/CTS is enabled.

I BLADE Contention Window Control Algorithm

We show the pseudo-code of BLADE contention window control algorithm in Alg. 1.

J Observation Interval Analysis

Assume that all Wi-Fi devices in the current Wi-Fi environment are collaboratively working to stabilize the MAR around the target value MAR_{tar}. We assume that the state of the channel being busy is represented by 1, and the state of the channel being idle is represented by 0. These busy or idle states of the channel over a period of time form an approximate i.i.d. sequence of Bernoulli random variables X_i, with success probability $MAR_{\text{tar}} = 0.15$. The sample mean over N_{obs} observations is:

$$\overline{X}_{N_{\text{obs}}} = \frac{1}{N_{\text{obs}}} \sum_{i=1}^{N_{\text{obs}}} X_i.$$

For $N_{\text{obs}} = 300$, the standard error (SE) of $\overline{X}_{300}$ is:

$$\text{SE}(\overline{X}_{300}) = \sqrt{\frac{0.15 \times 0.85}{300}} \approx 0.0206.$$

By the Chernoff bound for binomial distributions, the probability of deviation exceeding δ satisfies:

$$\mathbb{P}(|\overline{X}_{N_{\text{obs}}} - MAR_{\text{tar}}| \geq \delta) \leq 2\exp\left(-\frac{N_{\text{obs}}\delta^2}{3MAR_{\text{tar}}(1-MAR_{\text{tar}})}\right).$$

For $N_{\text{obs}} = 300$ and $\delta = 0.02$:

$$\mathbb{P}(|\overline{X}_{300} - 0.15| \geq 0.02) \leq 2e^{-0.314} \approx 1.462\%.$$

This confirms that the estimation error remains negligible with high probability. So $N_{obs} = 300$ is Sufficient.

	$MAR_{tar} = 0.1$ / IEEE	$MAR_{tar} = 0.25$ / IEEE	$MAR_{tar} = 0.35$ / IEEE	$MAR_{tar} = 0.5$ / IEEE
Avg. MAC Throughput (Mbps)	2.2 / 94.1	21.8 / 60.0	28.1 / 52.5	32.0 / 43.9
50th PPDU TX Delay (ms)	224.8 / 4.6	21.3 / 6.5	16.1 / 7.2	13.7 / 7.9
95th PPDU TX Delay (ms)	491.7 / 11.3	48.6 / 21.6	38.9 / 25.1	38.9 / 25.1
99th PPDU TX Delay (ms)	634.1 / 17.9	63.9 / 38.8	52.9 / 48.5	52.9 / 48.5
99.9th PPDU TX Delay (ms)	888.2 / 34.9	88.8 / 85.7	72.2 / 112.6	72.2 / 112.6

Table 6: The performance of BLADE coexisting with IEEE 802.11 standard contention control policy.

K Collision Probability in Wi-Fi Networks

Consider a scenario with N Wi-Fi devices, where each device has a transmission probability of τ in an arbitrary time slot and the transmission queue of each device remains non-empty. The collision probability, denoted as ρ, can be expressed as:

$$\rho = 1 - (1-\tau)^{N-1}. \tag{13}$$

Assume all Wi-Fi devices operate in the BE (Best Effort) queue, where the contention window (CW) doubles from $CW_{\min}$ to $CW_{\max}$ after each retransmission, up to a maximum of r retransmissions. Suppose x transmissions use $CW_{\min}$, and $x\rho^i$ transmissions use $CW_{\min}2^i$ for $0 \le i \le r$. The probability of transmitting with $CW_{\min}2^i$ is then given by:

$$P_i = \frac{x\rho^i}{\sum_{j=0}^{r} x\rho^j} = \frac{\rho^i}{\sum_{j=0}^{r} \rho^j}. \tag{14}$$

Thus, the transmission probability τ can be derived as:

$$\tau = \sum_{i=0}^{r} \frac{2P_i}{CW_{\min}2^i}. \tag{15}$$

By solving Eqn. 13, Eqn. 14 and Eqn. 15 simultaneously, the solution for ρ is obtained numerically using the bisection method within the range $(0,1)$, ensuring convergence to the unique solution for each N. Fig. 31 illustrates the variation in collision probability as the number of co-channel Wi-Fi devices increases. The analysis assumes that all Wi-Fi devices operate under the BE queue and maintain continuously non-empty transmission queues. The results indicate that when the number of co-channel Wi-Fi devices reaches 10, the collision probability exceeds 50%, highlighting the significant impact of device density on network performance.

L When MAR is Fixed, the Collision Probability is Constrained Below MAR

Consider a scenario where N WiFi devices continuously transmit data packets. Assuming that the contention window (CW) value for each device is $\omega - 1$, the attempt probability τ (i.e., the probability that any given WiFi device completes its random backoff and begins transmission at any slot-time [46] can be expressed as:

$$\tau = \frac{\omega}{\frac{\omega^2}{2}} = \frac{2}{\omega} \tag{16}$$

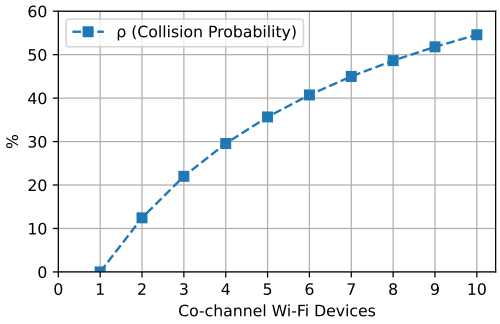


Figure 31: Collision probability vs. number of co-channel Wi-Fi devices (when the transmission queue remains non-empty).

The probability of a collision occurring when a WiFi device attempts transmission, denoted as ρ, is given by Eqn. 13. The MAR is defined as the probability that the channel is in a busy state, which corresponds to at least one WiFi device transmitting. This can be expressed as:

$$MAR = 1 - (1-\tau)^N \tag{17}$$

Given that $\tau = \frac{2}{\omega} \in (0,1)$ and is relatively close to zero, it follows that:

$$MAR = 1 - (1-\tau)^N > 1 - (1-\tau)^{N-1} = \rho \tag{18}$$

This implies that, when MAR is determined, the collision probability remains stable at a level below MAR.

Algorithm 1: BLADE contention window control.

Parameters

$N_{obs}, MAR_{tar}, MAR_{max}$ ▷ Default 300, 0.1, 0.35
CW_{min}, CW_{max} ▷ Default 15, 1023
$M_{inc}, M_{dec}, A_{inc}, A_{fail}$ ▷ Default 500, 0.95, 15, 5

Initialization

$N_{idle} \leftarrow 0$, $N_{tx} \leftarrow 0$, $first\_rtx \leftarrow True$,
$CW \leftarrow CW_{min}$, $CW_{fail} \leftarrow CW$

Func OnNewWNICState($state$, $duration$)
if $state = IDLE$ **then**
  $N_{idle} \leftarrow N_{idle} + duration/slot\_time$
if $state = BUSY$ **then**
  $N_{tx} \leftarrow N_{tx} + 1$

Func OnACK() ▷ Stable Control Policy
$CW \leftarrow CW_{fail}$ ▷ Restore the CW at previous failure
if $N_{idle} + N_{tx} < N_{obs}$ **then**
  return ▷ No enough samples
$MAR \leftarrow N_{tx}/(N_{tx} + N_{idle})$
if $MAR > MAR_{tar}$ **then**
  $CW \leftarrow CW + CW \cdot \max\{0, MAR - MAR_{max}\} +$
  $M_{inc}(\min\{MAR, MAR_{max}\} - MAR_{tar}) + A_{inc}$
else
  $CW \leftarrow \min\{M_{dec} - \frac{(1-M_{dec})(CW-CW_{min})}{CW_{max}-CW_{min}}, \frac{2MAR}{MAR_{tar}+MAR}\} \times CW$
$N_{idle} \leftarrow 0$, $N_{tx} \leftarrow 0$,
$CW_{fail} \leftarrow CW$, $first\_rtx \leftarrow True$

Func OnACKFailure() ▷ Fast Recovery from Collision
if $first\_rtx$ **then**
  $CW_{fail} \leftarrow CW + A_{fail}$ ▷ Increase and store CW
  $CW \leftarrow CW_{fail}/2$ ▷ Accelerated retransmission
  $first\_rtx \leftarrow False$ ▷ Only accelerate once