Supporting QUIC Data Flows over Consumer Electronic Devices: A Mobile Edge Computing-oriented Queuing Delay Control Policy

Yuanlong Cao, Jinquan Nie, Haopeng Zhang, Yirui Jiang, Jianmao Xiao, and Hong-Ning Dai, *Senior Member, IEEE*

***Abstract*—The Quick UDP Internet Connection (QUIC) protocol has been utilized in traditional cloud computing environments. However, in consumer electronic devices under edge cloud scenarios, QUIC may struggle to take advantage of its benefits due to its unique network characteristics and resource constraints. This problem arises when the network quality fluctuates or resources are insufficient to handle incoming data, causing rapid buffer expansion, significant delays, and potential packet loss. This study proposes a queue management algorithm inspired by the classical control theory of the Proportional-Integral-Differential (PID) control, which aims to support QUIC data flows to further enhance delay control and improve goodput. The algorithm adds differential operations to the traditional PI control to predict the error trend to respond to queue changes in advance. Combining expert experience in integral separation and queue error management, it makes the control strategy more relevant to the specific needs of real application scenarios. Simulation results demonstrate that the PID-Delay algorithm achieves an average goodput improvement of 1.98 times and reduces the standard deviation by 26.1% than the classical algorithm. In comparison to other delay algorithms, it exhibits an average 1.81 times increase in goodput and an 18.5% reduction in standard deviation.**

***Index Terms*—Mobile Edge Computing; QUIC Protocol; PID Control; Queue Management; Consumer Electronic Devices**

I. Introduction

The Quick UDP Internet Connection (QUIC) [1] protocol, having gained considerable traction within cloud-based environments, has showcased its myriad advantages: from ensuring minimal latency and robust congestion control to significantly enhancing the performance of HTTP/3 applications [2], [3]. With the rise of mobile edge computing, the focal point of computing and data processing is shifting from centralized data centers to the periphery of the network [4]. This evolution is characterized by the proliferation of data sources, encompassing everything from smart consumer electronic devices to sensors, thereby presenting unparalleled challenges to communication infrastructures [5]–[7].

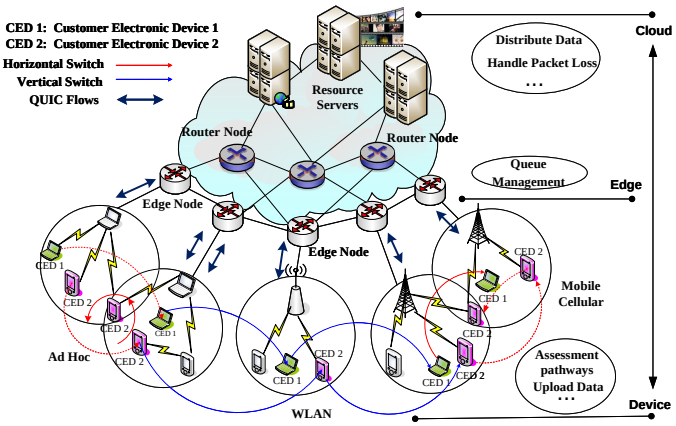

Fig. 1. A case of mobile edge computing scenario using QUIC protocol over consumer electronic devices.

Though QUIC demonstrates commendable efficacy in centralized cloud environments, its intrinsic characteristics may be emphasized, potentially posing challenges when implemented in edge computing scenarios. Currently, the primary research focus on QUIC revolves around its capabilities, including mitigating connection latency [8], improving data scheduling [9], [10], efficient multiplexing [11], and its inherent congestion control mechanisms [12], [13]. Nevertheless, its prowess in mobile edge computing contexts remains somewhat nebulous, particularly in pivotal domains like queue optimization and swift data transmission.

To delve deeper, certain QUIC functionalities, like connection migration [14] and 0-RTT connection resumption [15], which flourish in resource-abundant, stable cloud ecosystems, might inadvertently contribute to "bufferbloat" issues [16] in edge computing, especially on consumer electronic devices with constrained resources or in volatile network conditions. Confronted with these challenges, we contemplate a synergistic approach with the edge infrastructure to foster collective integration advancements during the assimilation of the QUIC protocol within the mobile edge computing paradigm.

Fig. 1 illustrates a transmission scenario involving consumer electronic devices, such as smartphones, tablets, and streaming devices, operating on the QUIC protocol within the framework

This work was supported by the National Natural Science Foundation of China under Grant No. 61962026, the Natural Science Foundation of Jiangxi Province under Grant No. 20224ACB202007, and the Jiangxi Provincial Department of Education Graduate Student Innovation Fund under Grant No. YC2023-S248. (Corresponding author: Hong-Ning Dai)

Yuanlong Cao, Jinquan Nie, Haopeng Zhang, and Jianmao Xiao are with the School of Software, Jiangxi Normal University, Nanchang 330022, China. (e-mail: ylcao@jxnu.edu.cn; jqnie@jxnu.edu.cn; hpzhang@jxnu.edu.cn; jm_xiao@jxnu.edu.cn)

Yirui Jiang currently holds a research fellow at the School of Water, Energy, and Environment, at Cranfield University. (e-mail: yirui.jiang@cranfield.ac.uk)

Hong-Ning Dai is currently an Associate Professor with the Department of Computer Science, at Hong Kong Baptist University, Hong Kong. (e-mail: henrydai@comp.hkbu.edu.hk)

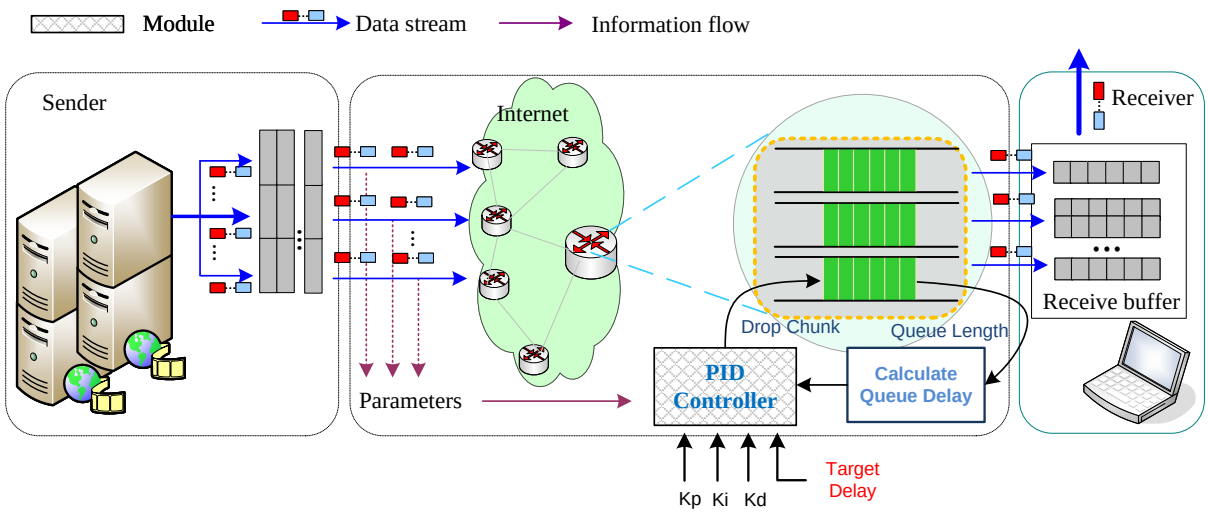

Fig. 2. PID control overview.

of mobile edge computing (MEC). Using mobile video transmission as a specific instance, mobile node terminals have the capability to initiate streaming sessions by establishing connections with resource servers facilitated through edge nodes. The substantial influx of data emanating from endpoint devices imposes significant strain on the infrastructure of the edge network. In this scenario, it becomes crucial to devise methodologies that offer differentiated support for services at the edge, deploy effective congestion control mechanisms, account for delays induced by queuing, and ultimately provide diverse Quality of Service (QoS) tiers for a range of applications and services [17].

To efficaciously address the challenges of "bufferbloat" and fortify the capability to manage queue delay, this study introduces a queue delay rectification strategy grounded in the Proportional-Integral-Differential Delay (PID-Delay) paradigm. Departing from traditional PI controls [18], this approach integrates differential operations and embraces comprehensive integral segregation, harmonized with nuanced expert insights into queue error management. By doing so, we can execute congestion control, decelerating the data dispatch rate proactively before the onset of packet overflow and consequent discards. This guarantees swift data transfer, achieving rapid convergence while minimizing queuing delays across various application contexts. The contributions of our research encompass the following:

- We craft the queue delay modulation methodology (PID-Delay) rooted in the PID control theorem tailored for the edge computing milieu. This approach thoroughly explores the complexities of delay error, skillfully navigating the interactions among the proportional, integral, and differential components to enhance the accuracy of queue delay management.
- We outline the mathematical model delineating the progression of the QUIC window within the AIMD framework [19]. Furthermore, we offer a comprehensive theoretical explanation and a critical assessment of the stability of the control strategy informed by PID control.
- We establish the queue delay regulation schema at the edge layer, proffering a robust framework for subsequent enactments of queue management at the terminal node. Concurrently, our comprehensive evaluations conducted via NS-3 simulation assays corroborate that the PID-Delay algorithm exhibits superior proficiency, optimizing both goodput efficacy and latency metrics.

The organization of the paper is as follows: Section II reveals the intricacies of the proposed scheme. Section III provides a theoretical exploration of both the window evolution paradigm and the dynamic queuing model. The performance evaluations are articulated in Section IV. The manuscript concludes with summarizing remarks and forward-looking perspectives in Section V and Section VI delves into the related work.

II. Scheme Design

Fig. 2 outlines a comprehensive approach for the surveillance and regulation of buffer queue latencies. This method incorporates feedback mechanisms grounded in classical control theory, particularly the utilization of PID controller. The principal aim of this approach is to enable real-time adjustments to the state attributes of the packets residing in the queue. This process begins with the computation of the current queue delay, which is determined by the length of the packets within the queue and their rate of departure. The PID controller calculates the probability of packet drop by assessing the difference between the preset target delay and the real delay. Depending on this computed drop probability, the system applies a relevant drop policy to decide whether the current packet should be discarded. This approach improves the effectiveness and accuracy of packet management within the network. For convenience, we begin by establishing the symbols defined in TABLE I.

A. PID-Delay Queue Control

The PID controller [20] represents a well-established feedback control technique widely employed in engineering control systems. In contrast, conventional queue management

TABLE I: NOTATIONS

Notations	Descriptions
$\tau(t)$	Queue delay function of the current time slice
τ_0	Target Delay
$O(t)$	Delay error function of the current time slice
$U(t)$	Control function of the current time slice
K_p	Proportional Factor
K_i	Integral Factor
K_d	Differential Factor
Z_1	Gain Factor #1
Z_2	Gain Factor #2
Z_3	Inhibitory Factor
M_1	Error Threshold #1
M_2	Error Threshold #2

approaches such as Drop-Tail and RED base their decisions primarily on queue length, offering simplicity and ease of implementation. However, these methods may lack precision in reflecting the true queue delay, particularly in scenarios with high packet arrival rates. To address this challenge, we introduce Equation (1), which provides an accurate means of calculating queue delay, thereby presenting a more adaptable, reliable, and efficient approach to queue control,

$$\tau(t) = \frac{QueueNBytes}{AvgDqRate}, \tag{1}$$

where $QueueNBytes$ represents the average number of packets in the queue and $AvgDqRate$ represents the average packet departure rate. Next, We further define the control error as the real-time queue delay $\tau(t)$ and a predetermined target delay τ_0 The difference between them, that is,

$$o(t) = \tau(t) - \tau_0, \tag{2}$$

where τ_0 sets a value for ideal convergence based on our network topology and network environment to ensure that the queue latency as a percentage of total latency is reduced.

$$U(t) = K_{\mathrm{p}} * o(t) + K_i * \int_0^{\mathrm{t}} o(t)dt + K_{\mathrm{d}} * \frac{do(t)}{dt}. \tag{3}$$

The QUIC network exhibits characteristics of nonlinearity and time variation, while its internal buffer queue aligns more closely with a discrete-time system. In PID control theory, Equation (3) represents its continuous form. To effectively implement a control strategy within a QUIC network, it becomes imperative to discretize Equation (3). This entails converting the continuous integral process into a discrete cumulative form and the continuous differential process into its discrete counterpart, resulting in the formulation of Equation (4) [21].

$$\begin{aligned} U(w\delta) = U((w-1)\delta) + K_p \cdot o(w\delta) + K_i \cdot (\sum_{w=1} o(w\delta)) \\ +K_d \cdot (o(w\delta) - o((w-1)\delta)). \end{aligned} \tag{4}$$

Considering the distinct features of the QUIC network, especially the discrete-time nature of its buffer queue, the continuous PID control strategy might face practical hurdles. Accumulation of $O(t)$ can result in increased computational complexity and potential overshooting. To address these issues, this study employs a differential incremental formulation, as depicted in Equation (5) [22].

$$\begin{aligned} \Delta U(t_i) = K_p \cdot (o(t_i) - o(t_{i-1})) + K_i \cdot o(t_i) \\ +K_d \cdot (o(t_i) - 2 \cdot o(t_{i-1}) + o(t_{i-2})). \end{aligned} \tag{5}$$

B. Error Adjustment

The implementation of PID control schemes necessitates refined modulation, particularly in scenarios of elevated network traffic, as documented in [23]. To cater to such demands, our framework integrates a triumvirate of coefficients: Z_1, Z_2, and Z_3. Here, Z_1 and Z_2 are operationalized as gain augmentation factors, whereas Z_3 functions as a damping factor within a stipulated boundary condition ($0 < Z_3 < 1 < Z_2 < Z_1$). The system architecture also demarcates two critical thresholds, defined as ($M_2 < M_1$). The judicious selection of the integral coefficient is central to the efficacy of PID control mechanisms. An inordinately high integral coefficient might precipitate overshooting or integral windup, conversely, an insubstantial coefficient may inadequately address persistent state errors. To circumvent these impediments, our research introduces a dynamic adjustment technique for the PID protocol that is contingent on the magnitude of the system's deviation.

Algorithm 1 provides a comprehensive breakdown of the error adjustment process. The following steps elucidate the algorithm's logic:

Step 1: Initialization of network parameters and assessment of the error trend between the current and previous moments.

Step 2: The current error's magnitude guides the initial division of the threshold dimension. Subsequently, the control strategy is fine-tuned based on the error trend. The specific strategies are delineated as follows:

1) When the condition $|O(t_i)| > M_1$ prevails, indicating that $O(t_i)$ is within a superior magnitude bracket, the corrective action on $\Delta U(t_i)$ should be executed by applying a gain coefficient scaled by Z_1. Simultaneously, it is imperative to negate the contribution of the integral component within the control adjustment mechanism.

2) In instances where $O(t_i) \cdot \Delta O(t_i) > 0$ coupled with $|O(t_i)| > M_2$, the error trajectory is ascending in absolute magnitude. Under these conditions, with $O(t_i)$ residing in an elevated range, the adjustment factor $\Delta U(t_i)$ should be scaled by a factor of Z_2 to facilitate a reversal in the error progression.

3) For scenarios where $O(t_i) \cdot \Delta O(t_i) < 0$, and $O(t_i) \cdot \Delta O(t_{i-1}) > 0$, the error exhibits a diminution in absolute value. At junctures where $|O(t_i)| > M_2$, it is prudent to modulate the $\Delta U(t_i)$ with a diminished Z_2 gain coefficient, concurrently nullifying the integral component's influence on the systemic dynamics.

4) When $O(t_i) \cdot \Delta O(t_i) < 0$, and $O(t_i) \cdot \Delta O(t_{i-1}) < 0$, the error is posited at a local extremum. Provided that $|O(t_i)| < M_2$, the corrective measure applied to $\Delta U(t_i)$ should incorporate the Z_3 multiplicative suppression factor, while concurrently abrogating the integral constituent's effect on the overarching system.

5) For all remaining conditions, a standard output should be sustained within the overarching PID control schema.

Algorithm 1: PID Error Adjustment Algorithm

Initialization of system parameters while setting $\Delta O(t_i) = O(t_i) - O(t_{i-1})$ and $\Delta O(t_{i-1})$ is the same as;
if *($|O(t_i)| > M_1$)* **then**
  The gain amplification of Z_1 is implemented for the growth of $\Delta U(t_i)$, while eliminating the effect of the integral component;
end
else if *($|O(t_i)|) > M_2$)* **then**
  if *($O(t_i) \cdot \Delta O(t_i) > 0$)* **then**
    The gain amplification of Z_2 is implemented for the growth of $\Delta U(t_i)$
  end
  else if *($O(t_i) \cdot \Delta O(t_i) < 0$ && $O(t_i) \cdot \Delta O(t_{i-1}) > 0$)* **then**
    The gain amplification of Z_2 is implemented for the growth of $\Delta U(t_i)$, while eliminating the effect of the integral component;
  end
end
else
  if *($O(t_i) \cdot \Delta O(t_i) < 0$&& $O(t_i) \cdot \Delta O(t_{i-1}) < 0$)* **then**
    The gain amplification of Z_3 is implemented for the growth of $\Delta U(t_i)$, while eliminating the effect of the integral component;
  end
end

III. MODEL ANALYSIS

Dynamic models of Transmission Control Protocol (TCP) behavior typically encompass the examination of fluid flow and stochastic differential equations (FDE) [24]. Fluid flow is harnessed to depict the data flow within the network, whereas stochastic differential equations capture and model the network's inherent uncertainty and randomness.

In dynamic modeling, both TCP and QUIC models include traditional features such as congestion control and flow control. The process of data transmission in QUIC can be accurately likened to the fluid flow within a network. The transmission of packets across a network can be likened to fluid flowing through a pipeline, where the network's topology and characteristics exert a substantial influence on the rate of packet transmission and congestion. As a result, the evolutionary model and queuing model of the QUIC window have been crafted, incorporating valuable insights gleaned from the dynamics model of TCP behavior [25].

A. Dynamic Model of QUIC

We assign labels to the QUIC streams as $i = 1, 2, \ldots, N$. It is assumed that the average queue length is derived through periodic sampling, utilizing a parameter ξ for calculating a weighted moving average, which is denoted as follows,

$$Q_i(t) = (1 - \xi) \cdot Q_i(t-1) + \xi \cdot Q_i(t), \tag{6}$$

where the parameter ξ, $0 < \xi < 1$. $R_i(t)$ denotes the round trip delay and is expressed in the following form,

$$R_i(t) = \aleph + \frac{Q_i(t)}{C}, \tag{7}$$

here, $\aleph$ is the fixed propagation delay, and $(Q_i(t))/C$ represents the queuing delay, where C is the link bandwidth capacity, and the host's processing delay is omitted for model simplification.

The variation in the QUIC window size is denoted as $\dot{W}(t)$. The differential equation for the expected window size and queue length is expressed in the following form,

$$\dot{W}(t) = \frac{1}{R(Q(t))} - \frac{W(t)W(t - R(t))}{2R(t)} p(Q(t - R(t))); \tag{8}$$

$$\dot{Q}_i(t) = \sum_{i=1}^{N} \frac{W(t)}{R(t)} - C. \tag{9}$$

The Equation (8) delineates the dynamic characteristics of the QUIC protocol in the congestion avoidance phase, specifically the congestion window size. In the meantime, Equation (9) encapsulates the unique characteristics stemming from the difference between packet arrival rates and link bandwidth capacities, expressed in a differential form reminiscent of Lindley's equation.

To ensure stability in the model described above, a small-signal linearization approach is employed to approximate the kinetic model of the window. This involves selecting an operating point represented as (W^*, p^*, Q^*) and using $W_R = W(t - R(t))$. The chosen operating point meets the following criteria.

Simplify the expression of the $R_i(t)$. The corresponding operating point is simplified and expressed as $R^*, N(t) \equiv N, C(t) \equiv C, \tau(t) \equiv \tau_0$,for the $\dot{W}(t) = 0$, $\dot{Q}(t) = 0$ can be obtained,

$$W^* = \frac{CR^*}{N}; (W^*)^2 p^* = 2; R^* = \frac{Q^*}{C} + \aleph. \tag{10}$$

The following are the relevant derivations of the differential equations for the window size and queue length.

$$\frac{\partial \dot{W}(t)}{\partial W} = \frac{\partial \dot{W}(t)}{\partial W_R} = \frac{-N}{(R^*)^2 C}; \quad \frac{\partial \dot{W}(t)}{\partial p} = \frac{-R^* C^2}{2N^2}. \tag{11}$$

$$\frac{\partial \dot{Q}(t)}{\partial W} = \frac{N}{R^*}; \quad \frac{\partial \dot{Q}(t)}{\partial Q} = \frac{\partial (\frac{NW(t)}{\aleph + \frac{Q(t)}{C}} - C)}{\partial Q} = -\frac{1}{R^*}. \tag{12}$$

The perturbation variables satisfy $\delta W = W - W^*$.$\delta Q = Q - Q^*$, establishing its differential equation,

$$\frac{d(\delta W(t))}{dt} = \frac{-N}{(R^*)^2 C} (\delta W(t) + \delta W_R) - \frac{R^* C^2}{2N^2} \delta p (t - R_0); \tag{13}$$

$$\frac{d(\delta Q(t))}{dt} = \frac{N}{R^*} \delta W(t) - \frac{1}{R^*} \delta Q(t). \tag{14}$$

Finding its corresponding Laplace transform yields that,

$$sW(s) = -\frac{N}{(R^*)^2 C} W(s) \left(1 + e^{-sR_0}\right) - \frac{R^* C^2}{2N^2} p(s) e^{-sR_0}; \tag{15}$$

$$sq(s) = \frac{N^*}{R^*}W(s) - \frac{1}{R^*}q(s). \tag{16}$$

The open-loop transfer function of the PID is known from Equation (3) as,

$$C_{PID}(s) = \frac{\frac{\alpha}{TC} + \frac{N\beta}{CS} + \frac{\gamma}{C}s}{s}. \tag{17}$$

Combining Equations (15)-(17) yields the final closed-loop transfer function as follows,

$$\mathcal{G}(s) \approx -\frac{\frac{C}{2N}\left(\frac{\alpha}{T} + \frac{N\beta}{S} + \gamma s\right)}{\left(s + \frac{N}{(R^*)^2 C}\right)\left(s + \frac{1}{R^*}\right)\left(1 + e^{-sR^*}\right)} \frac{e^{-sR^*}}{s}. \tag{18}$$

B. Stability Analysis

In the field of control system engineering, Bode plots and Nyquist plots serve as indispensable tools for scrutinizing the frequency response of a system [26]. These methods, especially in the development of PID control strategies, provide engineers with a simple way to evaluate and adjust the effectiveness of control techniques. In many instances, PID control structures are denoted as "controllers" or "compensators", with their primary objective being the preservation of closed-loop system stability.

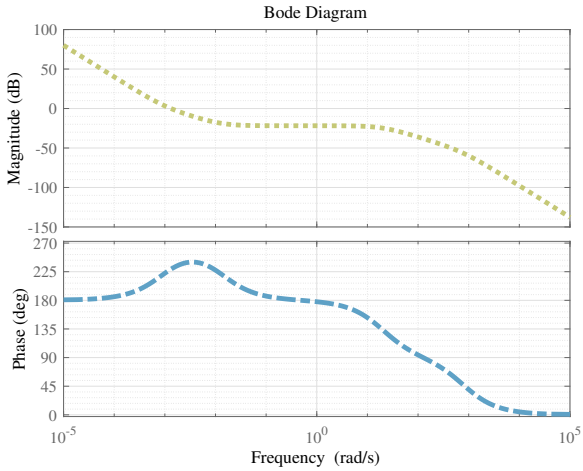

Fig. 3. Bode diagram.

Within the realm of frequency domain analysis, the Bode plot furnishes us with a dual logarithmic scale portrayal of the system's magnitude and phase response. In this context, gain margin and phase margin emerge as pivotal parameters for assessing system robustness. The gain margin illustrates the system's ability to handle variations in input amplitude or disturbances in the load. Meanwhile, the phase margin measures the maximum phase deviation that the system's response can tolerate without jeopardizing stability [27].

As illustrated in Fig. 3, it's easily noticeable that the system exhibits a notably generous gain margin, surpassing unity, along with a positive phase margin. These observations serve as compelling evidence of the system's stability. Moreover, the phase margin, measuring approximately 49 degrees, not only affirms the system's stability but also underscores its resilience to minor fluctuations in internal parameters or external disturbances.

The Nyquist stability theorem offers an insightful means of appraising the stability of linear time-invariant systems, obviating the need for direct computation of the closed-loop transfer function. This approach is particularly advantageous when we possess knowledge of the Nyquist trajectory of the open-loop transfer function. As delineated by using the axis-crossing version of Nyquist's stability theorem [28], the system's stability can be succinctly conveyed through the following equation, namely,

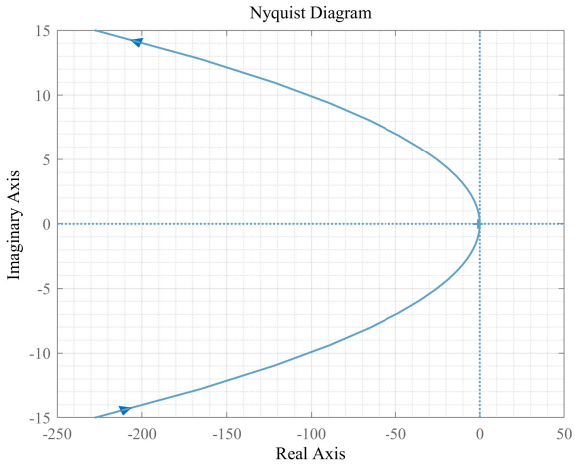

Fig. 4. Nyquist diagram.

$$P - Z = N, \tag{19}$$

where Z represents the count of poles residing in the right half of the complex plane for the closed-loop system, while P denotes the count of poles situated in the right half of the complex plane for the open-loop system. N signifies the number of counter-clockwise encirclements of the Nyquist trajectory around the point (-1,0). For a system to be deemed stable, the closed-loop transfer function should have no poles in the right half of the complex plane, resulting in $Z = 0$. Thus, the stability condition can be simplified to $P = N$. Upon examination of the open-loop transfer function, it becomes evident that all poles of the function are located in the left half of the complex plane, implying $P = 0$. When this observation is coupled with the insights from Fig. 4, where the Nyquist trajectory completes zero revolutions around (-1,0), it underscores the stability of the system.

IV. Performance Evaluation

This paper presents an experiment conducted on the NS-3 [29] platform to assess the efficacy of the PID-Delay algorithm in comparison to other queuing algorithms. The experiment utilized a particular network topology and various parameter configurations, as outlined in TABLE II. It's crucial to highlight that these parameters can be adjusted according to the specific network topology and performance criteria.

A. Experiment Establishment

1) Network Topology: As shown in Fig. 5, we have designed a multi-source, single-sink network structure where traffic sources are connected to the sink via access links and a shared bottleneck link. The access link employs a Drop-Tail queuing policy, whereas various queue management policies are implemented on the inter-gateway bottleneck link to enable performance comparisons. All sinks utilize the QUIC protocol for listening to traffic on specific ports, mimicking a real-time transmission scenario where a traffic source sends a substantial volume of data to the sink.

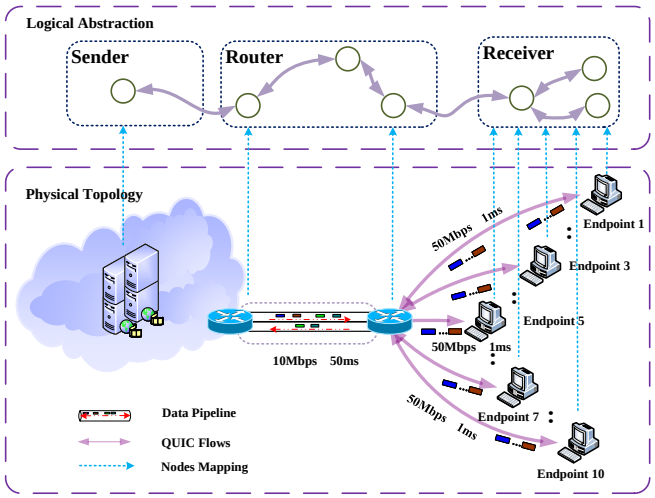

Fig. 5. Topology diagram.

TABLE II: Experimental Settings

Parameter	Value	Parameter	Value
Package Size	1440Bytes	K_p	1.3125
Access bandwidth	50Mbps	K_i	1.25
Access Delay	1ms	K_d	0.125
Bottleneck bandwidth	10Mbps	Z_1	1.5
Bottleneck Delay	50ms	Z_2	1.2
Buffer Size	4MB	Z_3	0.8
Loss Rate	1%	M_1	20ms
Queue Limit	1000p	M_2	10ms

2) Baseline Algorithms: Group A includes the Drop-Tail [30] and RED [31] algorithms, which are more oriented to traditional queue management methods, focusing on how to process and forward packets without paying much attention to the overall performance of the network.Group B algorithms include the PIE [32], CoDel [33], and Fq_CoDel [34] algorithms.The Group B algorithms, being more contemporary and intricate, strive to address prevalent issues in modern networks, such as buffer bloat, all the while ensuring optimal network performance and minimal latency.

Group A: DT (Drop-Tail), a fundamental queuing mechanism, adheres to FIFO (First-In-First-Out) queue principles by dropping packets when the queue reaches its full capacity.

Group A: RED (Random Early Detection) is an algorithm aimed at early congestion detection and preemptive action against queue overflow. It operates by randomly discarding packets at the front of the queue to mitigate congestion.

Group B: CoDel (Controlled Delay) addresses the notorious "bufferbloat" issue characterized by excessive delays due to the presence of extensive buffers. CoDel dynamically determines the appropriate time to discard a packet by observing the amount of time a packet spends in the queue. Once the queuing delay exceeds a predefined threshold, CoDel initiates congestion control measures to proactively manage the delay.

Group B: Fq_CoDel (Fair Queuing Controlled Delay) blends the principles of fair queuing with CoDel. It segregates incoming traffic into multiple queues and employs the CoDel algorithm for each queue. By adopting this approach, we guarantee fair distribution of network resources across all traffic streams, fostering low-latency service and implementing innovative congestion management.

Group B: PIE (Proportional Integral controller Enhanced) algorithm represents a cutting-edge queue management mechanism designed to alleviate congestion in contemporary networks. Drawing inspiration from control theory, PIE dynamically adjusts the probability of packet discarding based on real-time queue size and its growth trends. Utilizing a feedback-driven control approach empowers PIE to promptly adapt to changing network conditions, reducing delays and oscillations while effectively addressing buffer bloat. PIE's proactive congestion management approach makes it a pivotal solution for modern network scenarios, striving for peak goodput and minimal latency.

3) Experimental Design: In the delineation of network topology, we underscore the design of a single-source multi-destination network architecture, wherein traffic sources are linked to sink points via access links and interconnected with sink points through shared bottleneck links. Various queue management strategies are implemented on bottleneck links between gateways to facilitate performance comparisons.

We explicitly classify Group A and Group B algorithms, providing succinct introductions to each. Group A algorithms primarily emphasize traditional queue management methodologies, while Group B algorithms are more contemporary and intricate, aiming to address prevalent issues in modern networks such as buffer bloat, while simultaneously ensuring network performance and minimizing latency.

Concerning performance metrics, we opt for Goodput, Queue Delay, Round-Trip Time (RTT), and Loss Packets as the evaluative indicators for algorithm performance. These metrics comprehensively cover various facets of network performance, encompassing data transmission efficiency, latency, and queue delay, as well as network responsiveness, among others.

B. Performance Analysis

1) PID-Delay vs PIE: Fig. 6 provides valuable insights into the performance of the PIE and PID-Delay algorithms. These algorithms initially exhibit peak goodputs of 512.7 Kbps and 948.5 Kbps, respectively, during the initial 10 seconds of the experiment. This remarkable performance results from the synergistic influence of the proportional (P) and differential (D) parameters. Throughout the experiment, the PID-Delay

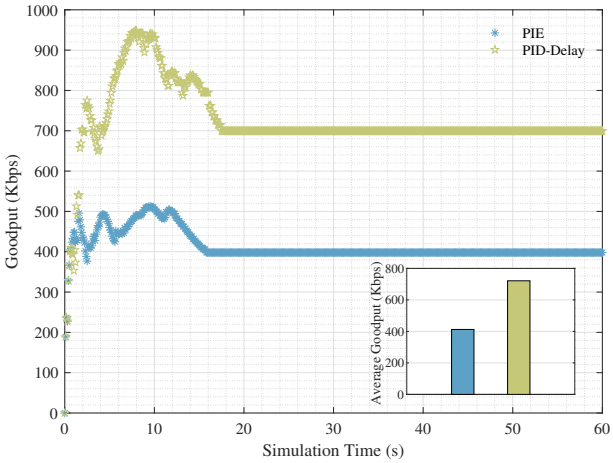

Fig. 6. Comparison Goodput of PIE and PID-Delay.

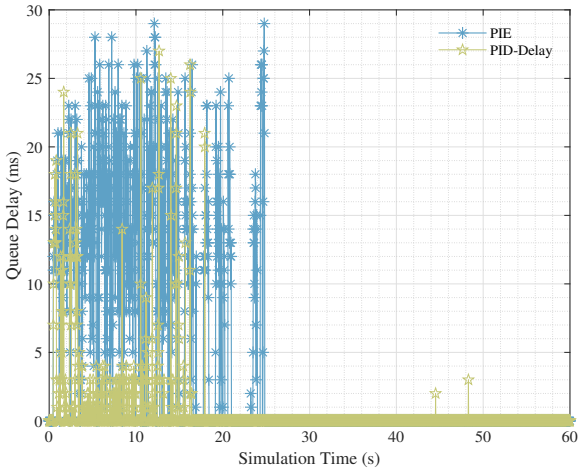

Fig. 7. Comparison Queue Delay of PIE and PID-Delay.

algorithm adaptively reacts to the cumulative error in the system. The integral term accumulates, leading the controller to generate a correction signal. This mechanism effectively rectifies extended periods of low goodput and congestion. Approximately 20 seconds into the experiment, the effective goodput of the PID-Delay algorithm stabilizes at approximately 699.4 Kbps. In comparison, the PIE algorithm exhibits an effective goodput of 397.9 Kbps after stabilization. The PID-Delay algorithm surpasses the PIE algorithm in both steady-state effective goodput and average effective goodput, demonstrating improvements of 1.76 times and 1.75 times, respectively. These results highlight the PID-Delay algorithm's capability, especially through its integral component, to consistently rectify errors, potentially resulting in improved stability and smoother operation. These advantages contribute to achieving higher goodput levels.

Fig. 7 illustrates the queue delay performance of the PIE and PID-Delay algorithms in the simulated experiment. In the first 20 seconds of the experiment, both algorithms demonstrate elevated queuing delay attributed to link congestion, with peak values of 29 ms and 27 ms, respectively. Subsequently, the PID-Delay algorithm demonstrates a significantly faster convergence of queuing delay variations compared to the PIE algorithm. A detailed analysis reveals that the average delay and standard deviation of the PIE algorithm amount to 4.14 ms and 7.25, respectively, while the PID-Delay algorithm boasts average delay and standard deviation values of 0.62 ms and 2.75. The PID-Delay algorithm exhibits a standard deviation 2.64 times smaller than that of the PIE algorithm. This improvement contributes to a more stable delay response, rendering it less susceptible to transient network perturbations when compared to the PIE algorithm.

2) PID-Delay vs classical Algorithms : Fig. 8 and Fig .9 provide a comprehensive overview of the performance of the PID-Delay algorithm in contrast to two classical algorithms within the context of the simulation experiments.

In Fig. 8, the PID-Delay algorithm stands out with remarkable effective goodput performance. The average effective goodput shows a substantial improvement, surpassing that of the DT and RED algorithms by 1.69 times and 2.28 times, respectively. The DT algorithm, which processes packets in the order of arrival, maintains a relatively stable goodput at approximately 400Kbps. During the first 20 seconds of the simulation experiment, the RED algorithm undergoes variations in goodput caused by random packet discards during congestion. Subsequently, as congestion alleviates, goodput rises, albeit with reduced stability. In contrast, the PID-Delay algorithm, benefiting from the combined impact of proportional, integral, and differential components, rapidly converges to an effective goodput of around 700Kbps.

Fig. 9 reveals that the PID-Delay algorithm, along with the RED algorithm, exhibits significant reductions in delay compared to the DT algorithm. The PID-Delay algorithm shows superior convergence relative to the RED algorithm. A detailed examination of the data indicates that the average latency for the PID-Delay, RED, and DT algorithms is 71.2 ms, 83.9 ms, and 242.4 ms, respectively. Furthermore, the standard deviation for these algorithms is 34.2, 42.6, and 50.6, respectively. The simpler queuing management approach employed by the DT algorithm results in higher latency, hovering around 240ms. The RED algorithm relies on the average queue size to gauge the need for congestion avoidance measures. While it demonstrates increased latency in the initial phases of the experiment, this latency diminishes significantly compared to DT. In contrast, the PID-Delay algorithm showcases a 15.1% reduction in latency compared to the RED algorithm, with lower jitter, further confirming its superiority. Additionally, the PID-Delay algorithm's standard deviation is reduced by 19.7% and 32.4% compared to the two classical algorithms, respectively.

3) PID-Delay vs Base Delay Algorithms : Fig. 10 and Fig. 11 illustrate the effective goodput and delay performance of the PID-Delay algorithm alongside two delay-based algorithms in the simulated experiments. As shown in Fig. 10, the PID-Delay algorithm significantly outperforms both the CoDel and Fq_CoDel algorithms, surpassing them by factors of 2.49 and 1.13, respectively. The CoDel algorithm, upon detecting that queue delay exceeds the predefined target latency, initiates

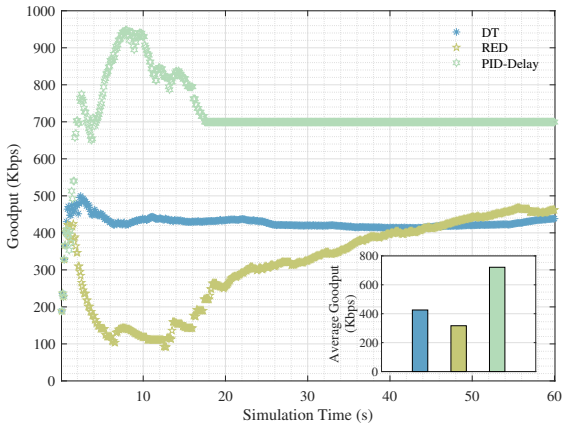

Fig. 8. Comparison goodput with classical algorithms

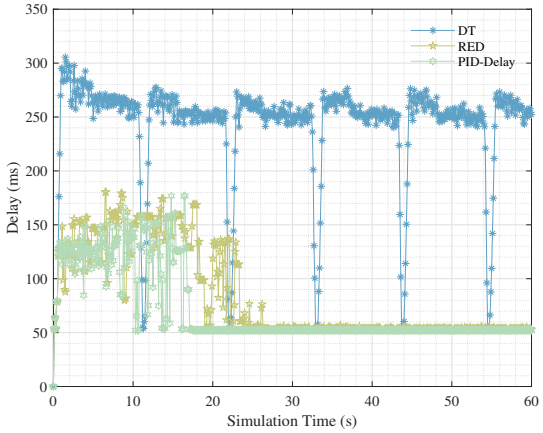

Fig. 9. Comparison delay with classical algorithms

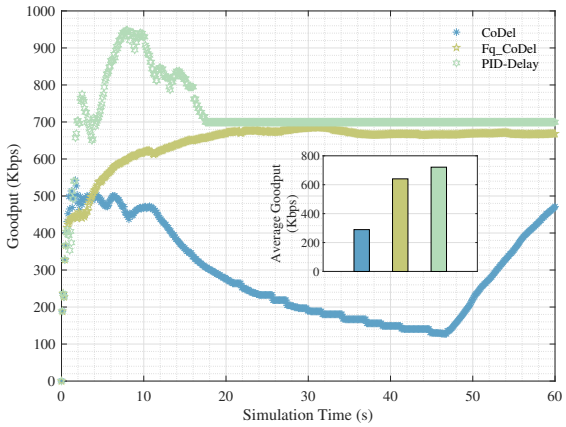

Fig. 10. Comparison goodput with delay algorithms

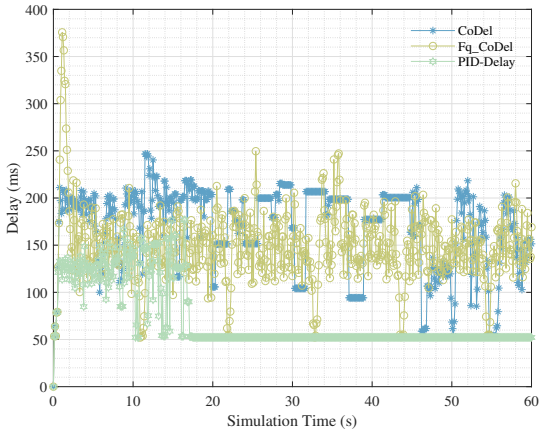

Fig. 11. Comparison delay with delay algorithms

active packet dropping, triggering congestion control mechanisms. Consequently, there is an initial decline in goodput. Nevertheless, as the algorithm adeptly handles the queue's latency on a lower level, goodput gradually restores to its usual state. The Fq_CoDel algorithm excels at automatically identifying and isolating new or pre-existing streams, safeguarding them from the behaviors of other streams. This adaptive stream isolation enhances overall goodput and delivers superior stability to the CoDel algorithm. After 20 seconds of experimentation, it achieves a steady-state goodput of 667Kbps.

Fig. 11 further highlights the PID-Delay algorithm's latency superiority over the CoDel and Fq_CoDel algorithms. An in-depth analysis of the data reveals that the average latency for the PID-Delay, CoDel, and Fq_CoDel algorithms is 71.2 ms, 169.4 ms, and 148.3 ms, respectively. The standard deviation for these algorithms is 34.2, 43.7, and 40.4. Importantly, the standard deviation of the PID-Delay algorithm is reduced by 15.3% and 21.7% compared to the two latency-based methods, respectively. This demonstrates that the PID-Delay algorithm is highly suitable for network link environments where low latency is a primary metric, offering excellent and stable performance in terms of latency.

As shown in Fig. 12 and Fig. 13, the performance of the PID-Delay algorithm is demonstrated with respect to the average RTT per Smart Endpoints of the other queue control algorithms (PIE, DT, RED, CoDel, and Fq_CoDel). Our further analysis of per Smart Endpoints (or Edge Consumer Electronic Devices) RTT shows that the average RTT of the PID-Delay algorithm is 196.8ms for 10 Smart Endpoints, and the average RTT of PIE, DT, RED, CoDel, and Fq_CoDel algorithms are 206.5 ms, 301.9 ms, 236.7 ms, 325.5 ms, and 228.2 ms, respectively. Yet, when examining Smart Endpoint 3, the RTT varies more significantly compared to the other Smart Endpoints. This is attributed to the fact that the PID-Delay algorithm is tailored for global optimization rather than local optimization. This implies that while the algorithm may surpass others in terms of overall performance, individual Smart Endpoints could encounter distinct network environments, congestion conditions, data stream sizes, and other variables.

Fig. 14 illustrates the number of packets lost during the data transmission process for several algorithms, representing the loss packets. It is evident from the graph that the PID-Delay and PIE algorithms exhibit periods of no packet loss, indicating their proficient packet control capabilities. However,

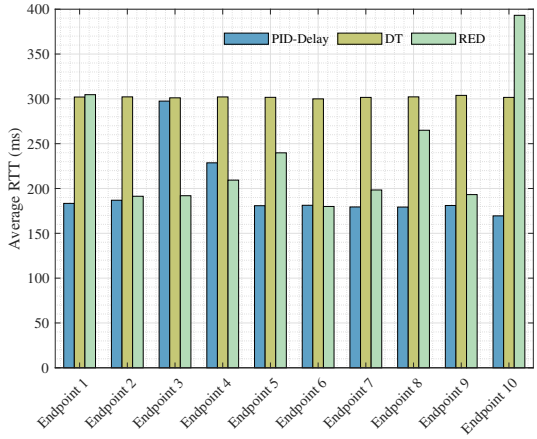

Fig. 12. Comparison average RTT with group B

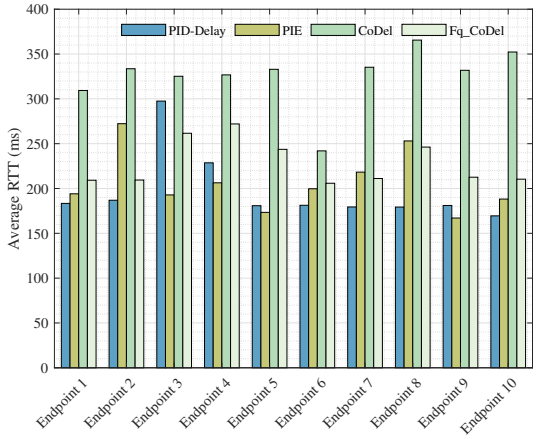

Fig. 13. Comparison average RTT with group C

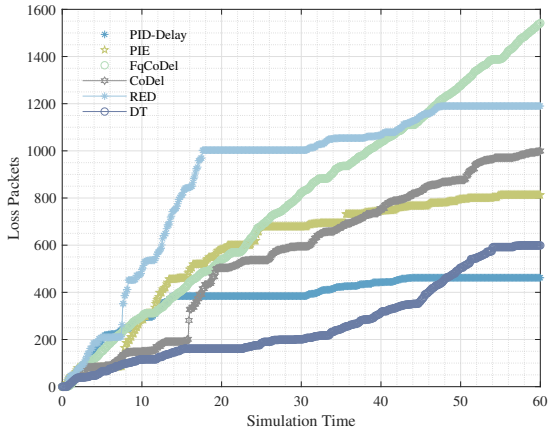

Fig. 14. Comparison Loss Packets

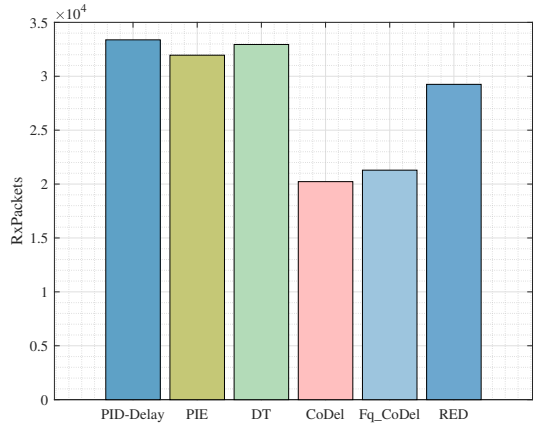

Fig. 15. Comparison RxPackets

the PIE algorithm struggles to maintain a stable period for an extended duration. Further analysis reveals that the average number of lost packets for PID-Delay and PIE algorithms is 374.9 and 581.9, respectively. Fig. 15 presents the number of packets successfully reaching their intended destination and being received, denoted as RxPackets. The RxPackets for PID-Delay, PIE, DT, RED, CoDel, and Fq_CoDel algorithms are 33380, 31955, 32949, 20225, 21290, and 29251, respectively. PID-Delay demonstrates superior packet reception capabilities.

V. Conclusion and Future Work

In this research, a PID control strategy derived from classical control theory is introduced to improve queue delay control and address buffer expansion issues. Our approach, in contrast to traditional PI control, integrates differential operations, accomplishes integral separation, and utilizes expert knowledge in queue error management. To thoroughly evaluate its performance, we conduct a comprehensive analysis of the system's magnitude and phase response using a control theory toolbox to ensure stability. Additionally, we perform experiments to compare its effectiveness with other widely used queue control algorithms, such as PIE, DT, RED, CoDel, and Fq_CoDel.

Our findings indicate that the PID-Delay algorithm achieves a stable goodput of 699.4 Kbps, nearly 1.76 times higher than the PIE algorithm. Not only does PID-Delay exhibit greater stability in queuing delay compared to PIE, but it also shows faster convergence. Statistical analysis reveals that, compared to classical algorithms, PID-Delay improves effective goodput by an average of 1.98 times and reduces its standard deviation by 26.1%. In comparison to other delay optimization algorithms, it achieves an average increase in effective goodput of 1.81 times and reduces its standard deviation by 18.5%. These experiments highlight the PID-Delay algorithm's capacity to deliver higher effective goodput and lower stable latency.

In our upcoming research pursuits, we aim to extend the existing PID-Delay algorithm framework to support multi-path terminal heterogeneous distributed network architectures [35]. Simultaneously, we are developing endpoint solutions to tackle the challenges of the "last-hop" segment in the network [36]. Additionally, we are actively involved in devising a transport strategy rooted in cloud-side cooperative computing with reinforcement learning [37]. This strategy aims to integrate various intelligent computing capabilities to sense real-time network dynamics, ensuring network friendliness to meet the dynamic and intricate demands of real-time networks.

VI. Related Work

The IETF QUIC Working Group has played a pivotal role in standardizing and developing the QUIC protocol. In May 2021, they released RFC 9000 [36], marking the protocol's formal standardization. Additional RFCs, such as RFC 9368 [37], focus on compatibility mechanisms between QUIC clients and servers. RFC 9308 [38] discusses the protocol's applicability, particularly its impact on application protocol development and deployment. RFC 9312 [39] provides guidance for network management in handling encrypted QUIC traffic. Lastly, RFC 9221 [40] introduces extensions to QUIC, enhancing its capabilities for sending and receiving unreliable datagrams. Together, these efforts support the deployment and advancement of QUIC in network communication.

Since the formal standardization of the QUIC protocol, its performance has undergone extensive measurement and validation. K. McMillan et al. [41] proposed a comprehensive set of formal specifications and test methodologies for the QUIC protocol. Additionally, T. Shreedhar et al. [42] investigated the protocol's performance under various workloads, including web storage and video. D. Madariaga et al. [43] examined QUIC network traffic in Android mobile applications and assessed its performance across different workloads, such as web storage and video.K. Hou et al. [44] proposed a QUIC protocol enhancement based on QUIC network traffic analysis. They also introduced a QUIC-based service acceleration and security optimization scheme for Serverless Networks, validating its performance through testing. J. Dizdarević et al. [45] examined HTTPS within the "Continuum" of IoT Cloud/Edge, while also evaluating the HTTP/QUIC (HTTP/3) protocol's scalability and latency in the same IoT Cloud/Edge "Continuum" setting.

Meanwhile, both domestic and international scholars have undertaken a series of innovative research endeavors concerning the optimization of the QUIC protocol. These efforts primarily encompass the development of congestion control mechanisms, flow control and scheduling strategies, retransmission algorithms, and related areas. Q. Coninck et al. [46] devised a QUIC framework that facilitates pluggable deployments, enabling users to flexibly (plug-and-play) select and modify the congestion control algorithm. F. Chiariotti et al. [47] proposed a QUIC multi-stream scheduling optimization algorithm is proposed. This algorithm intelligently incorporates the characteristics of Correlated Data Flows (CDF) to maximize the Value of Information (VoI) of received data, ultimately improving transmission efficiency. E. Volodina et al. [48] presented advancements in the credit-based Flow Control mechanism within the QUIC protocol stack, aiming to optimize the performance of QUIC multiplexing. G. Sinha et al. [49] applied cross-layer design to the QUIC protocol stack and constructed the "application layer (HTTP)" + "transport layer (QUIC)" and "transport layer (QUIC)" + "physical layer (SINR)" respectively. The cross-layer collaborative communication framework is constructed, and the data flow transmission scheduling is carried out accordingly to improve the transmission service quality. D. Bhat et al. [50] optimize the Quality of Experience (QoE) of Adaptive Bitrate Streaming (ABS) using the QUIC retransmission mechanism

In the realm of multipath QUIC technology optimization, current research primarily centers on enhancing the multipath QUIC scheduling mechanism. H. Wu et al. [51] devised a dynamic multipath QUIC data scheduler capable of learning and adapting to transmission characteristics, such as packet loss and delay, across heterogeneous network links. X. Shi et al. [52] proposed a priority-based multipath QUIC stream scheduling mechanism to mitigate inter-stream blocking issues by prioritizing streams based on their importance. X. Shi et al. [53] applied the concept of stream prioritization to the multipath QUIC data scheduling mechanism and designed a multipath QUIC scheduling policy with time-critical streams (time-sensitive streams) prioritized. Moreover, X. Shi et al. [54] introduced a multipath QUIC scheduling mechanism prioritizing page-critical rendering streams, thereby reducing loading and rendering times through optimized scheduling and transmission of critical data streams. Furthermore, G. Choudhary et al. [55] developed the MultiPipe QUIC (MP-QUIC) extension protocol, which incorporates two distinct data scheduling algorithms: the Round Robin-based MultiPipe QUIC scheduling algorithm and the cross-layer aware MultiPipe QUIC data scheduling algorithm.

In the landscape of both domestic and international research, studies concerning the QUIC protocol predominantly concentrate on congestion control mechanisms, flow control, and multipath scheduling strategies. However, there exists a notable dearth of discussion and research regarding the integration of queue management algorithms and associated mechanisms. This paper endeavors to address this gap by leveraging the QUIC protocol in conjunction with classical control theory methods to delve extensively into queue management algorithms.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant No. 61962026, the Natural Science Foundation of Jiangxi Province under Grant No. 20224ACB202007, and the Jiangxi Provincial Department of Education Graduate Student Innovation Foundation under Grant No. YC2023-S248.

References

[1] A. Langley, J. Iyengar, J. Bailey, J. Dorfman, J. Roskind, J. Kulik, et al., "The QUIC Transport Protocol", in Proc. of *the Conference of the ACM Special Interest Group on Data Communication*, pp.183-196, 2017.

[2] T. Shreedhar, R. Panda, S. Podanev, and V. Bajpai, "Evaluating QUIC Performance Over Web, Cloud Storage, and Video Workloads," *IEEE Transactions on Network and Service Management*, vol.19, no.2, pp.1366-1381, Jun. 2022.

[3] M. R. Kanagarathinam, K. M. Sivalingam and S. Lee, "A Neural Network-Based Network Selection for QUIC to Enrich Gaming in NextGen Wireless Network," *IEEE Transactions on Consumer Electronics*, vol. 70, no. 1, pp. 4536-4547, Feb. 2024.

[4] T. Zhang et al., "QoE-Driven Data Communication Framework for Consumer Electronics in Tele-Healthcare System,"*IEEE Transactions on Consumer Electronics*, vol. 69, no. 4, pp. 719-733, Nov. 2023.

[5] J. -H. Syu, J. C. -W. Lin, G. Srivastava and K. Yu, "A Comprehensive Survey on Artificial Intelligence Empowered Edge Computing on Consumer Electronics," *IEEE Transactions on Consumer Electronics*, vol. 69, no. 4, pp. 1023-1034, Nov. 2023.

[6] X. Zhou, W. Liang, K, Yan, W. Li, K. I. Wang, J. Ma, and Q. Jin, "Edge-Enabled Two-Stage Scheduling Based on Deep Reinforcement Learning for Internet of Everything," *IEEE Internet of Things Journal*, vol.10, no.4, pp.3295-3304, Feb. 2023.

[7] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys and Tutorials*, vol.19, no.4, pp.2322-2358, Fourth quarter 2017.

[8] E. Sy, T. Mueller, M. Moennich, and H. Federrath, "Accelerating QUIC's Connection Establishment on High-Latency Access Networks," in Proc. of *IEEE International Performance Computing and Communications Conference (IPCCC)*, London, UK, pp.1-8, 2019.

[9] F. Michel, A. Cohen, D. Malak, Q. De Coninck, M. Médard, and O. Bonaventure, "FlEC: Enhancing QUIC With Application-Tailored Reliability Mechanisms," *IEEE/ACM Transactions on Networking*, vol.31, no.2, pp.606-619, Apr. 2023.

[10] Y. Xing et al., "A Stream-Aware MPQUIC Scheduler for HTTP Traffic in Mobile Networks," *IEEE Transactions on Wireless Communications*, vol.22, no.4, pp.2775-2788, Apr. 2023.

[11] E. Volodina, and E. P. Rathgeb, "Flow Control in the Context of the Multiplexed Transport Protocol QUIC," in Proc. of *IEEE Conference on Local Computer Networks (LCN)*, Sydney, NSW, Australia, pp.473-478, 2020.

[12] H. Haile, K. -J. Grinnemo, S. Ferlin, P. Hurtig, and A. Brunstrom, "WIP: Leveraging QUIC for a Receiver-driven BBR for Cellular Networks," in Proc. of *IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Pisa, Italy, pp.252-255, 2021.

[13] Y. Liu, Z. Yang, Y. Peng, T. Bi, and T. Jiang, "Bandwidth-Delay-Product-Based ACK Optimization Strategy for QUIC in Wi-Fi Networks," *IEEE Internet of Things Journal*, vol.10, no.20, pp.17635-17646, Oct. 2023.

[14] Y. Yan, and Z. Yang, "When QUIC's Connection Migration Meets Middleboxes A case study on mobile Wi-Fi hotspot," in Proc. of *IEEE Global Communications Conference (GLOBECOM)*, Madrid, Spain, pp.1-6, 2021.

[15] X. Cao, S. Zhao, and Y. Zhang, "0-RTT Attack and Defense of QUIC Protocol," in Proc. of *IEEE Global Communications Conference Workshops (GLOBECOM WKSHPS), Waikoloa*, HI, USA, pp.1-6, 2019.

[16] Y. Gong, D. Rossi, C. Testa, S. Valenti, and M. D. Täht, "Fighting the bufferbloat: On the coexistence of AQM and low priority congestion control," in Proc. of *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Turin, Italy, pp.411-416, 2013.

[17] J. Zhang, X. Zhong, Z. Wan, Y. Tian, T. Pan, and T. Huang, "RCC: Enabling Receiver-Driven RDMA Congestion Control With Congestion Divide-and-Conquer in Datacenter Networks," *IEEE/ACM Transactions on Networking*, vol.31, no.1, pp.103-117, Feb. 2023.

[18] C. V. Hollot, V. Misra, D. Towsley, and Weibo Gong, "Analysis and design of controllers for AQM routers supporting TCP flows," *IEEE Transactions on Automatic Control*, vol.47, no.6, pp.945-959, Jun. 2002.

[19] L. Cai, X. S. Shen, J. W. Mark, and J. Pan, "QoS support in Wireless/Wired networks using the TCP-Friendly AIMD protocol," *IEEE Transactions on Wireless Communications*, vol.5, no.2, pp.469-480, Feb. 2006.

[20] Z. Pan, F. Dong, J. Zhao, L. Wang, H. Wang, and Y. Feng, "Combined Resonant Controller and Two-Degree-of-Freedom PID Controller for PM-SLM Current Harmonics Suppression," *IEEE Transactions on Industrial Electronics*, vol.65, no.9, pp.7558-7568, Sept. 2018.

[21] Y. Xu, H. Deng, P. Zhang, and J. Yang, "Tuning PI/PID active queue management controllers supporting TCP/IP flows," in Proc. of *International Conference on Communications (ICC)*, Circuits and Systems, Hong Kong, China, pp.630-634, 2005.

[22] S. Tzafestas, and N. P. Papanikolopoulos, "Incremental fuzzy expert PID control," *IEEE Transactions on Industrial Electronics*, vol.37, no.5, pp.365-371, Oct. 1990.

[23] T. Qi, and H. Wang, "PID sliding mode controller design and application to active queue management," in Proc. of *Chinese Control Conference (CCC)*, Chengdu, China, pp.6917-6922, 2016.

[24] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of aqm routers supporting TCP flows with an application to red," in Proc. of *ACM International Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, pp.151–160, 2000.

[25] C. V. Hollot, V. Misra, D. Towsley, and W. bo Gong, "A control theoretic analysis of red," in Proc. of *IEEE International Conference on Computer Communications (INFOCOM)*, pp.1510–1519, 2001.

[26] L. Miao, J. Z. Xin, and B. Zhao, "Revisit Nyquist-Bode Stability Criteria for Power Electronic System with Non-minimum Phase System," in Proc. of *IEEE International Future Energy Electronics Conference (IFEEC)*, Singapore, pp.1-6, 2019.

[27] N. Munro, "The systematic design of PID controllers using Nyquist stability," in Proc. of *European Control Conference (ECC)*, Porto, Portugal, pp.528-533, 2001.

[28] G. F. Franklin, J. D. Powell, A. Emami-Naeini, Feedback Control of Dynamic Systems, (3rd Ed), Addison-Wesley,1994.

[29] The NS3 Simulator. Accessed: Apr. 10, 2021. [Online]. Available: "https://www.nsnam.org."

[30] O. Lemeshko, O. Yevsyeyeva, and S. Garkusha, "QoS ensuring scheme for telecommunication networks with Tail Drop and RED mechanisms," in Proc. of *International Conference on the Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*, Lviv, Ukraine, pp.214-216, 2013.

[31] L. Xue, "Simulation of Network Congestion Control Based on RED Technology," in Proc. of *International Conference on Computational and Information Sciences*, Shiyang, China, pp.1497-1500, 2013.

[32] R. Pan et al., "PIE: A lightweight control scheme to address the bufferbloat problem," in Proc. of *IEEE International Conference on High Performance Switching and Routing (HPSR)*, Taipei, Taiwan, pp.148-155, 2013.

[33] N. Khademi, D. Ros, and M. Welzl, "The new AQM kids on the block: An experimental evaluation of CoDel and PIE," in Proc. of *IEEE International Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Toronto, ON, Canada, pp.85-90, 2014.

[34] T. Høiland-Jørgensen, "Analyzing the Latency of Sparse Flows in the FQ-CoDel Queue Management Algorithm," *IEEE Communications Letters*, vol.22, no.11, pp.2266-2269, Nov. 2018.

[35] X. Zhou, Q. Yang, Q. Liu, W. Liang, K. Wang, Z. Liu, J. Ma, and Q. Jin, "Spatial-Temporal Federated Transfer Learning with Multi-Sensor Data Fusion for Cooperative Positioning," *Information Fusion*, vol. 105, May 2024.

[36] Y. Cao, D. Yu, L. Zeng, Q. Liu, F. Wu, X. Gui, M. Huang, "Towards Efficient Parallel Multipathing: A Receiver-centric Cross-layer Solution to Aid Multipath TCP," in Proc. of *IEEE International Conference on Parallel and Distributed Systems (ICPADS)*, pp.790-797, 2019.

[37] X. Zhou, X. Zheng, X. Cui, J. Shi, W. Liang, Z. Yan, L. T. Yang, S. Shimizu, and K. Wang, "Digital Twin Enhanced Federated Reinforcement Learning with Lightweight Knowledge Distillation in Mobile Networks," *IEEE Journal on Selected Areas in Communications (JSAC)*, vol.41, no.10, pp.3191-3211, Oct. 2023.

[38] J. Iyengar, M. Thomson, "QUIC: A UDP-Based Multiplexed and Secure Transport," RFC 9000, IETF, May 2021.

[39] D. Schinazi, E. Rescorla, "Compatible Version Negotiation for QUIC," IETF RFC 9368, 2023.

[40] M. Kühlewind, B. Trammell, "Applicability of the QUIC Transport Protocol," RFC 9308, IETF, Sep. 2022.

[41] M. Kühlewind, B. Trammell, "Manageability of the QUIC Transport Protocol," RFC 9312, IETF, Sep. 2022.

[42] T. Pauly, E. Kinnear, D. Schinazi, "An Unreliable Datagram Extension to QUIC," RFC 9221, IETF, Apr. 2022.

[43] K. McMillan, L. Zuck, "Formal specification and testing of QUIC," in Proc. of *ACM SIGCOMM*, pp.227-240, Aug. 2019.

[44] T. Shreedhar, R. Panda, S. Podanev, V. Bajpai, "Evaluating QUIC Performance over Web, Cloud Storage and Video Workloads," *IEEE Transactions on Network and Service Management*, vol.19, no.2, pp. 1366-1381, Jun. 2022.

[45] D. Madariaga, L. Torrealba, J. Madariaga, J. Bermúdez, J. Bustos-Jiménez, "Analyzing the Adoption of QUIC From a Mobile Development Perspective," in Proc. of *ACM SIGCOMM Workshop on the Evolution, Performance, and Interoperability of QUIC*, pp.35-41, 2020.

[46] K. Hou, S. Lin, Y. Chen, V. Yegneswaran, "Accelerate and secure serverless networks with QUIC," in Proc. of *ACM 17th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, pp.477-478, Dec. 2021.

[47] J. Dizdarević, A. Jukan, "Experimental Benchmarking of HTTP/QUIC Protocol in IoT Cloud/Edge Continuum," in Proc. of *IEEE International Conference on Communications (ICC)*, pp.1-6, Jun. 2021.

[48] Q. Coninck, F. Michel, M. Piraux, F. Rochet, T. Wilson, A. Legay, O. Pereira, O. Bonaventure, "Pluginizing QUIC," in Proc. of *ACM SIGCOMM*, pp.59-74, Aug. 2019.

[49] Q. Coninck, F. Michel, M. Piraux, F. Rochet, T. Wilson, A. Legay, O. Pereira, O. Bonaventure, "Pluginizing QUIC," in Proc. of *ACM International Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (ACM SIGCOMM)*, pp.59-74, Aug. 2019.

[50] F. Chiariotti, A. Deshpande, M. Giordani, K. Antonakoglou, A. Zanella, "QUIC-EST: A QUIC-Enabled Scheduling and Transmission Scheme to Maximize VoI with Correlated Data Flows," *IEEE Communications Magazine*, vol.59, no.4, pp.30-36, Apr. 2021.
[51] G. Sinha, M. Kanagarathinam, S. Jayaseelan, G. Choudhary, "CQUIC: Cross-Layer QUIC for Next Generation Mobile Networks," in Proc. of *IEEE Wireless Communications and Networking Conference (WCNC)*, pp.1-8, May 2020.
[52] D. Bhat, R. Deshmukh, M. Zink, "Improving QoE of ABR Streaming Sessions through QUIC Retransmissions," in Proc. of the 26th ACM International Conference on Multimedia (MM), pp.1616-1624, Oct. 2018.
[53] H. Wu, Ö. Alay, A. Brunstrom, S. Ferlin, G. Caso, "Peekaboo: Learning-Based Multipath Scheduling for Dynamic Heterogeneous Environments," *IEEE Journal on Selected Areas in Communications*, vol.38, no.10, pp.2295-2310, Oct. 2020.
[54] X. Shi, L. Wang, F. Zhang, B. Zhou, Z. Liu, "PStream: Priority-Based Stream Scheduling for Heterogeneous Paths in Multipath-QUIC," in Proc. of *29th International Conference on Computer Communications and Networks (ICCCN)*, pp.1-8, 2020.
[55] X. Shi, L. Wang, F Zhang, Z Liu, "FStream: Flexible Stream Scheduling and Prioritizing in Multipath-QUIC," in Proc. of *IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*, pp.1-8, 2019.
[56] X. Shi, F. Zhang, Z. Liu, "PriorityBucket: A Multipath-QUIC Scheduler on Accelerating First Rendering Time in Page Loading," in Proc. of *ACM International Conference on Future Energy Systems*, pp.572-577, Jun. 2020.
[57] G. Choudhary, M. Kanagarathinam, H. Natarajan, K. Arunachalam, S. Jayaseelan, G. Sinha, D. Das, "Novel MultiPipe QUIC Protocols to Enhance the Wireless Network Performance," in Proc. of *IEEE Wireless Communications and Networking Conference (WCNC)*, pp.1-7, 2020.

Yuanlong Cao received the B.S. degree in computer science and technology from Nanchang University, China, in 2006, the M.S. degree in software engineering from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2008, and the Ph.D. degree in communication and information system from the Institute of Network Technology, BUPT, in 2014. He was an Intern/Software Engineer with BEA TTC, IBM CDL, and DT Research, Beijing, China, from 2007 to 2011. He is currently a Professor with the School of Software, Jiangxi Normal University, Nanchang, China. His research interests include multimedia communications, network security and next-generation Internet technology. Dr. Cao serves as the Editor for KSII Transactions on Internet and Information Systems. He has served as the Lead Guest Editor for Mobile Networks and Applications, Intelligent Automation and Soft Computing, International Journal of Distributed Sensor Networks, Electronics, Wireless Communications and Mobile Computing, Security and Communication Networks, International Journal of Digital Multimedia Broadcasting, and the Guest Editor for IEICE Transactions on Information and Systems, Computers, Materials & Continua. He serves as the Co-General Chair of the 12th EAI International Conference on Mobile Networks and Management (EAI MONAMI 2022), the Co-General Chair of the 4th EAI International Conference on Data and Information in Online Environments (EAI DIONE 2023), and the Co-Chair of IEEE 9th World Forum on Internet of Things Workshop on Advancements in Metaverse and IoT, respectively. He has also served as the Technical Reviewer for several journals, including the IEEE Transactions on Industrial Informatics, IEEE Transactions on Network and Service Management, IEEE Transactions on Cognitive Communications and Networking, etc.

Jinquan Nie received the B.S. degree in the School of Artificial Intelligence and Big Data at Hefei University, China, in 2022.

He is currently pursuing an M.S. degree in Management Science and Engineering at Jiangxi Normal University. His research interests include multipath transmission control protocol, cybersecurity and information management and systems.

Haopeng Zhang received the B.S. degree in the School of Information Engineering from the Shaoguan University, China, in 2022.

He is currently pursuing an M.S. degree in Management Science and Engineering at Jiangxi Normal University. His research interests include multipath transmission, next-generation network protocols, and Internet-related technologies.

Yirui Jiang is a Data Scientist, Artificial Intelligence, and Machine Learning expert. With a passion for extracting actionable insights from complex data sets, she brings a wealth of knowledge and expertise to the field of data science. Equipped with a deep understanding of advanced statistical analysis, data modeling, and predictive analytics, she possesses the skills necessary to transform raw data into meaningful and valuable information. Her expertise in machine learning algorithms, neural networks, and deep learning techniques empowers her to develop innovative solutions and make accurate predictions in diverse domains. She successfully applied her expertise to a wide range of projects, she is well-versed in tackling real-world challenges and driving data-driven decision-making. She thrives in interdisciplinary teams, leveraging her expertise to contribute to cross-functional projects and foster a collaborative environment. With a commitment to staying at the forefront of the ever-evolving field of data science, she continuously expands their knowledge through ongoing learning and exploration of emerging technologies and techniques. Her dedication to professional growth ensures her remains at the cutting edge of the field and enables her to deliver innovative solutions to complex problems.

Jianmao Xiao received the Ph.D. from the College of Intelligence and Computing, Tianjin University. He is an assistant professor at Jiangxi Normal University, is the deputy director of the Jiangxi Provincial Engineering Research Center of Blockchain Data Security and Governance. He is a member of the CCF TCSC. His main research interests include blockchain, service computing, intelligent software engineering. Dr. Xiao has authored more than 30 high-level academic papers, served as MONAMI 2022 Web Chair and ICSS 2022 PC Member. He also served as a reviewer for many domestic and international high-level journals and conferences in related fields.

Hong-Ning Dai (Senior Member, IEEE) received the Ph.D. degree in computer science and engineering from the Department of Computer Science and Engineering, The Chinese University of Hong Kong. Currently, he is an Associate Professor with the Department of Computer Science, at Hong Kong Baptist University, Hong Kong. His current research interests include the Internet of Things, big data, and blockchain technology. He has served as an Editor for Computer Communications (Elsevier), Connection Science (Taylor & Francis), and IEEE Access, and a Guest Editor for IEEE Transactions on Industrial Informatics, IEEE Transaction Emerging Topics in Computing, and IEEE Open Journal of The Computer Society .