PPO-Driven Reinforcement Learning Congestion Control Under High-BDP Wide-Area Deployment: A Scaling Analysis

Ali Ghermezian, Kirill Karpov, Dmitry Kachan, Veronika Kirova and Eduard Siemens
Department of Electrical, Mechanical and Industrial Engineering, Anhalt University of Applied Sciences, Bernburger Str. 55, 06366 Köthen, Germany
ali.ghermezian@hs-anhalt.de, kirill.karpov@hs-anhalt.de, dmitry.kachan@hs-anhalt.de, veronika.kirova@hs-anhalt.de, eduard.siemens@hs-anhalt.de

Keywords: Reinforcement Learning, Congestion Control, Proximal Policy Optimization (PPO), High Bandwidth-Delay Product (BDP), Wide-Area Networks (WAN), Aurora.

Abstract: Reinforcement learning (RL) has recently been explored as an adaptive alternative to hand-designed congestion control, yet most reported evaluations remain confined to simulated or moderate-bandwidth environments. This paper studies the scaling behavior of an Aurora-style Proximal Policy Optimization (PPO) congestion control policy when it is deployed on a real high-bandwidth, high bandwidth-delay product (BDP) wide-area network (WAN) path with a 10 Gbps interface budget. The purpose is not to claim a new state-of-the-art controller, but to identify how a simulator-trained PPO policy behaves when transferred to multi-gigabit operation. We integrate the policy through a user-space PCC shim and analyze transport-level logs, including achieved receive rate, loss dynamics, rate oscillations, and available round-trip time (RTT) indicators. The results show a persistent gap between nominal link capacity and achieved single-flow goodput: the analyzed run reaches a mean receive rate of 49.6 Mb/s and a peak of 438.0 Mb/s, corresponding to 0.50% average and 4.38% peak utilization of the 10 Gbps budget. Additional TCP CUBIC and TCP BBR baselines on the same path reached up to 9.91 Gb/s and 9.72 Gb/s with four flows, confirming that the route itself can sustain multi-gigabit throughput. The rate distribution is heavy-tailed, with short high-rate episodes followed by overshoot-collapse dynamics, bursty loss, and transient delay inflation. These findings indicate that scale-aware training, more robust reward normalization, and lower-overhead pacing are needed before PPO-driven congestion control can generalize reliably to high-BDP WAN deployments.

1 INTRODUCTION

Congestion control is a fundamental component of transport protocols, responsible for achieving high link utilization while preventing persistent queue buildup and instability [1]. In high-speed and multi-gigabit wide-area networks (WANs), this problem becomes substantially more challenging due to large bandwidth-delay products (BDP), amplified feedback delays, and increased sensitivity to transient queue dynamics [2]. Classical loss-based algorithms may underutilize available capacity in such environments, while delay-based and control-theoretic approaches attempt to stabilize queue evolution under high-speed conditions.

For example, Karpov et al. analyze performance enhancement strategies for multi-gigabit WANs and emphasize the scaling challenges introduced by large BDP and delayed feedback [3]. Mareev et al. further investigate PID-based congestion control for high-speed IP networks, showing that explicit control-theoretic mechanisms can improve stability compared to heuristic schemes, yet remain sensitive to parameter tuning and network variability [4]. These limitations have motivated the exploration of data-driven approaches, including reinforcement learning (RL), as an alternative to manually designed congestion control logic [5].

RL-based congestion control frames rate adaptation as a sequential decision-making problem: the sender observes recent network statistics, selects a control action, receives a reward that reflects throughput, loss, and delay, and then updates or applies a learned policy. Aurora formulates this problem as a partially observable Markov decision process and uses Proximal Policy Optimization (PPO), a policy-gradient RL algorithm designed to improve a policy while limiting excessively large policy updates [1],

[6]. PPO is attractive for congestion control because it can optimize continuous rate-control actions and can learn non-linear responses to noisy transport measurements. However, these same learned responses may become fragile when the operating scale differs substantially from the training environment.

Published Aurora evaluations report promising results in controlled and moderate-bandwidth environments. However, Aurora's experimental validation has largely been confined to Mbps-scale simulations and emulation, typically below 128 Mb/s [1]. Modern backbone and research networks operate at multi-gigabit rates, where BDP amplification fundamentally alters control dynamics: feedback becomes relatively slower, queueing transients become more costly, and exploratory rate changes can induce amplified oscillations. The behavior and generalization properties of simulator-trained PPO congestion control under real high-BDP multi-gigabit WAN conditions remain insufficiently understood [7].

In this study, we empirically evaluate an Aurora-style PPO congestion control policy deployed over a real wide-area path under a 10 Gbps interface budget. Rather than claiming universal performance improvements, we focus on characterizing utilization limits, heavy-tailed rate behavior, overshoot-collapse dynamics, and instability patterns observed in practice. Our analysis highlights structural constraints, including reward scaling sensitivity, delayed feedback amplification, and user-space pacing overhead. Baseline throughput measurements are reported for path-capacity validation, while full multi-flow fairness under simultaneous coexistence remains future work.

2 METHOD AND METRICS

This section describes the PPO-driven control architecture, the observations used by the policy, the decision interval, the reward signal, and the measurement metrics used for the scaling analysis.

2.1 Control Architecture

We evaluate an Aurora-style congestion control policy integrated through a user-space PCC shim [8]. The sender periodically observes recent transport statistics and outputs a continuous control signal that multiplicatively adjusts the current sending rate.

The implementation follows Aurora's abstraction:

$$R_{t+1} = R_t \cdot \exp(\alpha \cdot a_t), \quad (1)$$

where Rt, measured in Mb/s, is the current sending rate, at is the continuous action output by the policy, alpha is a fixed delta-scale factor, and the rate is clipped within predefined minimum and maximum bounds. Rate limits in our deployment were configured as Rmin = 500 Mb/s and Rmax = 5 Gb/s. These bounds were adjusted during scaling experiments to explore high-BDP behavior. The controller operates entirely in user space through the PCC-RL shim interface.

2.2 Observation Space

At each decision interval, the policy receives a bounded history of recent transport statistics. The observation vector includes estimated throughput over the last interval, packet loss rate, an RTT sample, sending rate, and delivery ratio or acknowledged bytes. The implementation uses a fixed history length of H = 10 intervals. Thus, the observation vector is a concatenation of the last 10 measurement tuples, forming a temporally stacked state representation. All features are normalized using running statistics consistent with the Aurora training implementation, using mean-variance normalization.

2.3 Decision Interval

The control decision is executed at fixed monitoring intervals corresponding to the PCC measurement epoch. In our deployment, the decision interval is approximately one PCC monitor interval, or about 100-200 ms depending on traffic dynamics. This interval length directly affects stability under high BDP, because delayed feedback amplifies multiplicative rate updates.

2.4 Reward Function

The original Aurora reward formulation maximizes a utility combining throughput and delay penalties:

$$r_t = \text{throughput}_t - \beta \cdot \text{delay}_t - \gamma \cdot \text{loss}_t. \quad (2)$$

In our implementation, the reward is computed inside the shim using throughput-based utility consistent with the PCC-RL training plugin:

$$r_t = U(\text{throughput}_t, \text{loss}_t), \quad (3)$$

where the delay contribution is limited due to unavailable RTT precision in receiver logs. For analysis purposes, we report a normalized reward proxy derived from logged throughput and inferred loss dynamics.

2.5 Normalization

Feature normalization follows Aurora's training convention: throughput is normalized relative to a recent mean, loss is represented as a bounded fraction, the action output is constrained to a finite range, and the reward is normalized before PPO updates. However, when transferring to multi-gigabit WAN conditions, absolute throughput scales exceed those seen during simulation training, potentially causing reward scaling mismatch.

2.6 Training and Checkpoint Regime

The evaluated policy was initialized from a pre-trained PPO checkpoint trained in simulated environments under moderate bandwidth regimes. During deployment, online PPO updates were enabled, model checkpoints were periodically saved, and no retraining with high-BDP domain randomization was performed. Thus, the experiments evaluate transfer and scaling behavior rather than re-optimized high-speed training.

2.7 Measurement Metrics

From receiver-side logs, we extract receive rate in Mb/s as the primary goodput proxy, aggregate sent and lost counters, and per-interval rate statistics. Because RTT values in available logs are constant placeholders in part of the dataset, latency-based analysis is limited. Therefore, tail behavior is characterized through mean receive rate, standard deviation, p50, p95, p99, and empirical cumulative distribution function (CDF). These metrics capture burstiness and heavy-tail dynamics under high-BDP conditions.

3 EXPERIMENTAL SETUP

Experiments were conducted on a Linux-based server equipped with a 10 Gbps network interface and a 24-core CPU architecture. The congestion control policy, implemented as an Aurora-style PPO controller, was executed in user space using Stable-Baselines PPO2 and interfaced through the PCC shim driver. Traffic generation and feedback were collected via sender logs and receiver-side CSV exports.

The evaluation focuses on real wide-area conditions with a high-BDP path. Performance statistics were derived exclusively from transport-level logs, including per-interval throughput, RTT indicators, cumulative bytes sent, and loss counters. Receiver logs report a peak single-flow throughput of approximately 438 Mb/s in the analyzed run, while multi-flow experiments occasionally exceeded 1 Gb/s in aggregate throughput when multiple solver instances utilized separate CPU cores.

Single-flow execution was observed to operate under user-space pacing constraints, while multi-flow configurations leveraged parallel CPU scheduling to increase aggregate throughput. Baseline RTT measurements extracted from receiver logs ranged around a few milliseconds under low-queue conditions, with significant spikes observed during rate overshoot events.

All experiments were performed using identical policy checkpoints and fixed hyperparameters to isolate deployment effects from training variability. No kernel-level congestion control modifications were applied; all control logic operated entirely in user space.

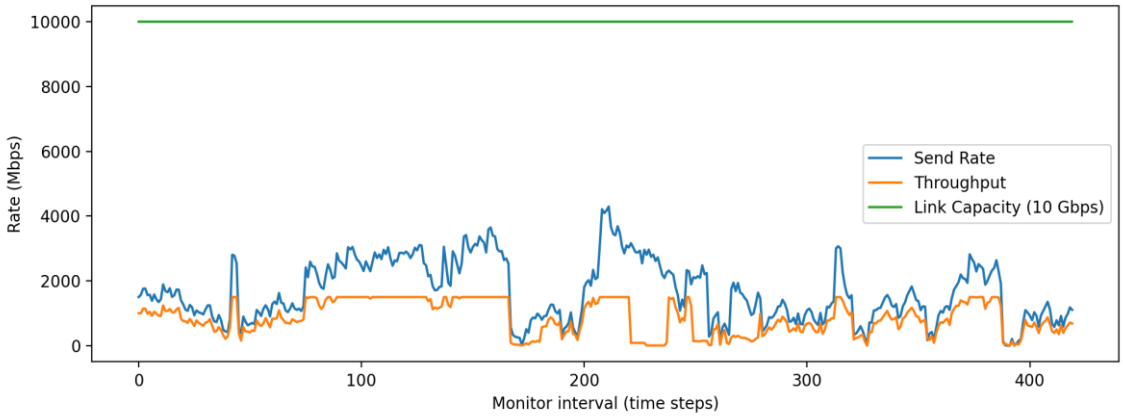

Figure 1: Send rate and achieved throughput over time under a 10 Gbps link capacity.

To address baseline performance on the same physical path, we additionally measured TCP CUBIC and TCP BBR using iperf3 on the Canada-to-Cuba WAN path, with destination 14.14.14.16 and a 10 Gbps nominal interface budget. We also measured PCC/Vivace without RL and RL-controlled PCC over the same user-space PCC deployment using four flows on ports 9001-9004. These baseline measurements were used to distinguish physical path capacity from user-space PCC/RL scaling limitations.

To address baseline performance on the same physical path, we additionally measured TCP CUBIC and TCP BBR using iperf3 on the Canada-to-Cuba WAN path, with destination 14.14.14.16 and a 10 Gbps nominal interface budget. We also measured PCC/Vivace without RL and RL-controlled PCC over the same user-space PCC deployment using four flows on ports 9001-9004. These baseline measurements were used to distinguish physical path capacity from user-space PCC/RL scaling limitations.

4 RESULTS

This section reports the observed utilization, temporal rate behavior, and distributional properties of the achieved receive rate. The purpose is to make the empirical scaling behavior explicit rather than to present the controller as an optimized production implementation.

4.1 Achieved Utilization and Temporal Behavior

Across the analyzed receiver log, the mean receive rate is 49.6 Mb/s with a standard deviation of 90.1 Mb/s and a peak of 438.0 Mb/s. Relative to a 10 Gbps budget, this corresponds to an average utilization of 0.50% and peak utilization of 4.38%. Figure 1 shows pronounced variability and intermittent spikes, consistent with scaling challenges under high-BDP transfer. The time series indicates that the controller can occasionally raise the sending rate, but these increases are not sustained long enough to maintain high utilization.

4.2 Tail Behavior of Achieved Rate

The distribution of achieved receive rate is heavy-tailed: p50 = 18.0 Mb/s, p95 = 334.2 Mb/s, and p99 = 388.8 Mb/s, as shown in Figure 2. Such tail spikes indicate that short-lived bursts occur, but sustained high utilization is not achieved in this run. This is consistent with a generalization gap between simulator training regimes and real 10 Gbps WAN conditions [9].

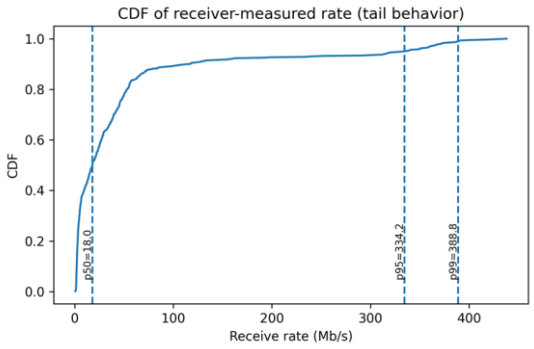

Figure 2: CDF of receiver-measured rate; vertical markers show p50, p95, and p99.

4.3 Loss and Delay Indicators

Figure 3 compares loss-rate evolution with available RTT indicators. Aggressive rate increases correlate with loss bursts and transient RTT inflation, suggesting queue buildup and delayed feedback amplification. Because the RTT field is not fully reliable in all receiver logs, these delay observations should be interpreted qualitatively. Nevertheless, the synchronized pattern of rate excursions, loss bursts, and delay spikes supports the conclusion that high-BDP feedback delay contributes to instability.

4.4 Baseline Comparison on the Canada-to-Cuba WAN Path

To determine whether the low utilization was caused by the network path itself or by the user-space PCC/RL stack, we conducted additional baseline tests on the same Canada-to-Cuba WAN path. Table 1 summarizes TCP CUBIC, TCP BBR, PCC/Vivace without RL, RL-controlled PCC, and the original analyzed RL run. Single-flow TCP results provide a reference for the original analyzed RL run, while four-flow TCP results match the multi-flow PCC/Vivace and RL-PCC measurements and demonstrate available path capacity under parallel flows. The TCP baselines confirm that the path can sustain multi-gigabit throughput: four-flow CUBIC and BBR achieved 9.91 Gb/s and 9.72 Gb/s, respectively. In contrast, four-flow PCC/Vivace without RL and RL-controlled PCC remained near 1 Gb/s. Figure 4 visualizes the same goodput comparison.

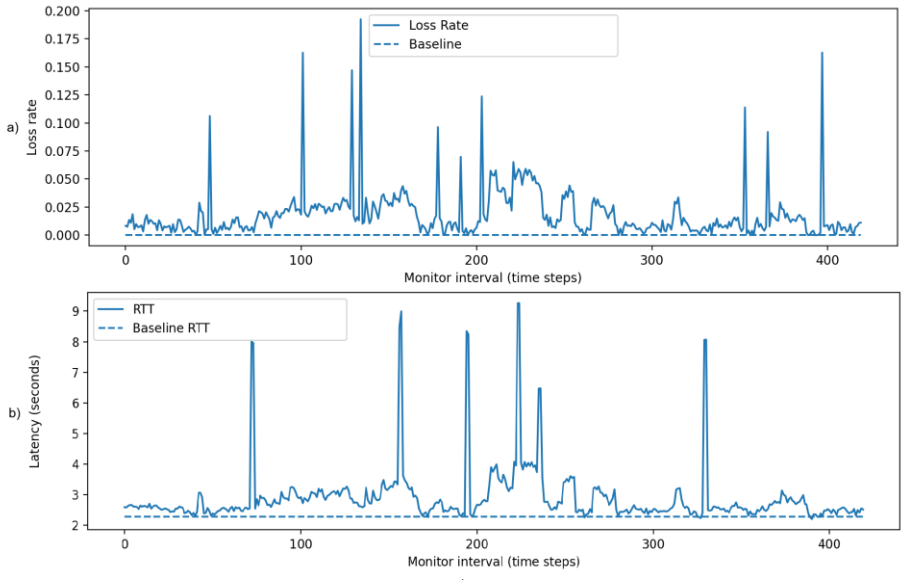

Figure 3: Loss rate and RTT evolution over time. Aggressive rate increases correlate with loss bursts and transient RTT inflation, reflecting queue buildup and delayed feedback dynamics.

Table 1: Baseline comparison on the Canada-to-Cuba WAN path.

Method	No.	Source	Mean	Peak	RTT	Loss/retrans.	Util.
TCP CUBIC	1	iperf3	4.57 Gb/s	N/A	N/A	2,460 retrans.	45.7%
TCP BBR	1	iperf3	8.72 Gb/s	N/A	N/A	692,138 retrans.	87.2%
TCP CUBIC	4	iperf3	9.91 Gb/s	N/A	N/A	5,422 retrans.	99.1%
TCP BBR	4	iperf3	9.72 Gb/s	N/A	N/A	48,116 retrans.	97.2%
PCC/Vivace	4	PCC	0.987 Gb/s	1.251 Gb/s	20.29 ms	4 lost pkt.	9.87%
RL-PCC	4	PCC/RL	0.967 Gb/s	1.022 Gb/s	20.20 ms	280 lost pkt.	9.67%
Original RL	1	Orig. data	0.0496 Gb/s	0.438 Gb/s	Limited	Bursty loss	0.50%

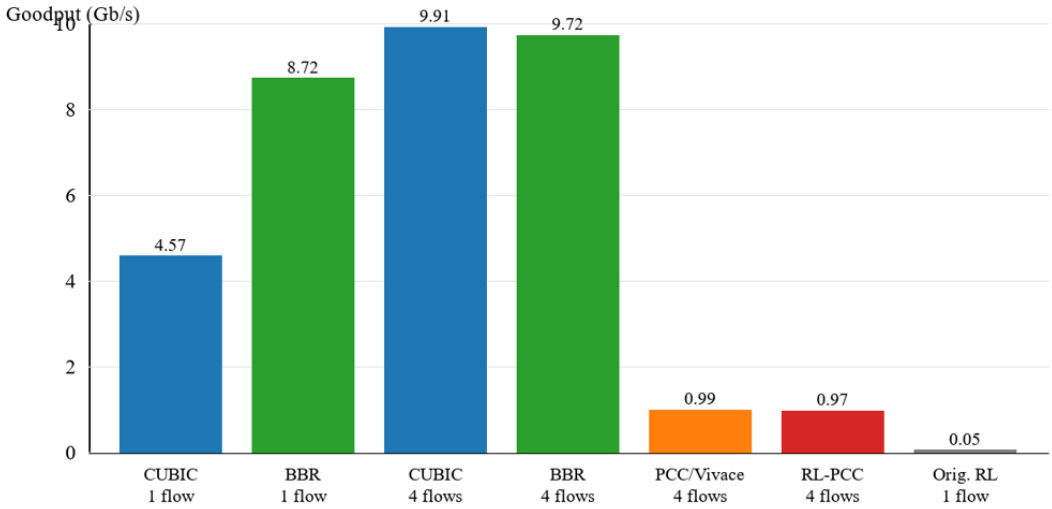

Figure 4: Mean goodput comparison on the same Canada-to-Cuba WAN path. TCP CUBIC and BBR confirm that the path can sustain multi-gigabit throughput, while user-space PCC/Vivace and RL-controlled PCC remain near 1 Gb/s.

These results show that the physical WAN path is not the primary bottleneck. Instead, the observed performance gap is consistent with limitations in the user-space PCC/RL data path, packet tracking, pacing precision, and feedback-driven control loop under high-BDP operation. The RL-controlled PCC configuration did not outperform PCC/Vivace without RL in this test; it achieved slightly lower mean goodput and higher packet loss.

5 DISCUSSION

The empirical results reveal a substantial gap between the nominal interface capacity and the achieved goodput in single-flow operation. Receiver-side measurements indicate that throughput remains far below the available link budget, while the CDF analysis confirms that high-rate episodes are rare and short-lived rather than sustained.

The send-rate time series in Figure 1 exhibits pronounced overshoot-collapse behavior. Rapid multiplicative rate increases are followed by abrupt drops, indicating unstable adaptation dynamics under high-BDP conditions. This instability is further supported by Figure 3, where aggressive rate excursions correlate with transient loss bursts and RTT spikes, reflecting queue buildup and delayed feedback amplification.

These observations suggest that policies trained under moderate-bandwidth simulation regimes may not generalize seamlessly to multi-gigabit WAN environments, particularly when deployed in real-network settings where measurement noise, scheduling variability, and implementation overhead are present [8]. Prototype experimentation frameworks such as Iroko [10] have been proposed to facilitate controlled evaluation of reinforcement learning congestion control policies prior to deployment. However, translating such prototypes to high-BDP wide-area environments introduces additional system-level constraints, including large in-flight data volumes and amplified feedback delay, which increase sensitivity to exploratory rate updates and promote oscillatory behavior.

The implementation-level constraints are important. A user-space controller must schedule monitoring, policy inference, action application, and packet pacing outside the kernel data path. At multi-gigabit rates, CPU scheduling jitter, timer granularity, socket-buffer behavior, and packet pacing precision can limit achievable throughput even when the learned policy selects aggressive rates. The observation that multi-flow runs improve aggregate throughput by using multiple CPU cores suggests that computation and pacing overhead contribute to the single-flow ceiling, although the dataset does not isolate these effects experimentally.

The additional baseline comparison strengthens this interpretation. TCP CUBIC and TCP BBR reached 9.91 Gb/s and 9.72 Gb/s in four-flow operation on the same path, confirming that the physical route and 10 Gbps interface budget can support near-line-rate throughput. By contrast, PCC/Vivace without RL and RL-controlled PCC remained close to 1 Gb/s. Therefore, the utilization ceiling observed for the learning-based deployment should be attributed primarily to user-space PCC/RL implementation and control-loop scaling constraints rather than to insufficient path capacity.

While multi-flow execution improves aggregate throughput through multi-core utilization, per-flow instability patterns remain visible in loss and latency traces. Fairness under coexistence with conventional TCP variants was not fully evaluated beyond the baseline throughput comparison and remains an open question for future study. Systematic evaluations of learning-based congestion control highlight the importance of quantitative fairness, efficiency, and responsiveness metrics when assessing deployment viability [9].

Overall, the combined evidence from rate dynamics, loss behavior, RTT evolution, and distributional analysis indicates that both control-level sensitivity and deployment-level constraints contribute to the observed utilization ceiling in high-BDP wide-area environments.

6 LIMITATIONS

The present study is limited to a single high-BDP wide-area deployment and does not explore systematic variations in RTT, cross-traffic intensity, or alternative bottleneck configurations. As a result, the observed instability patterns may depend on the specific path characteristics considered here.

The performance analysis is primarily derived from transport-level logs. Although sufficient for identifying macro-level dynamics, the available measurements restrict fine-grained reconstruction of transient loss events. Moreover, RTT measurements are not fully available with sufficient precision across all logs, so latency analysis remains limited and should be interpreted as indicative rather than definitive.

The implementation operates entirely in user space. User-space pacing and per-interval

computation may introduce additional scheduling overhead that constrains achievable throughput, particularly at multi-gigabit rates. CPU scheduling, timer resolution, packet pacing precision, and kernel versus user-space data-path differences should be isolated in future experiments.

Finally, the added CUBIC and BBR measurements provide a path-capacity baseline, but they do not constitute a full coexistence or fairness study under mixed traffic. A controlled comparison with simultaneous competing flows would still be necessary to quantify fairness and responsiveness under shared-bottleneck conditions.

7 CONCLUSIONS

This work empirically examined the scaling behavior of an Aurora-style PPO congestion control deployment over a real high-BDP wide-area path with a 10 Gbps interface budget. The measurements show that single-flow utilization remains significantly below nominal capacity and is characterized by oscillatory rate dynamics, bursty loss behavior, and transient RTT inflation.

The throughput distribution further indicates that high-rate episodes are short-lived rather than sustained, suggesting that policy adaptation becomes increasingly sensitive in large bandwidth-delay environments. Taken together, these results imply that transferring reinforcement learning-based congestion control from moderate-bandwidth training regimes to multi-gigabit WAN deployments requires greater robustness to delayed feedback, scale-aware reward normalization, and improved implementation support for precise high-rate pacing.

Future investigations should extend the analysis to simultaneous coexistence experiments with CUBIC and BBR, explicit fairness evaluation, richer RTT and queue measurements, and architectural optimizations that reduce user-space pacing overhead.

8 ACKNOWLEDGMENTS

This work was supported by the European Regional Development Fund (ERDF/EFRE) and the State of Saxony-Anhalt within the programme Sachsen-Anhalt WISSENSCHAFT Forschung und Innovation (EFRE) 2021-2027, project AI-RMDT (grant no. ZS/2023/12/182323). We acknowledge support by the German Research Foundation (Deutsche Forschungsgemeinschaft, DFG) and the Open Access Publishing Fund of Anhalt University of Applied Sciences.

9 REFERENCES

[1] N. Jay, N. T. Li and B. Hariharan, “A deep reinforcement learning perspective on internet congestion control,” in Proceedings of the 36th International Conference on Machine Learning (ICML), Long Beach, CA, USA, 2019, pp. 3050-3059, [Online]. Available: https://proceedings.mlr.press/v97/jay19a/jay19a.pdf?utm_source=chatgpt.com.

[2] K. Winstein and H. Balakrishnan, “TCP ex machina: Computer-generated congestion control,” ACM SIGCOMM Computer Communication Review, vol. 43, no. 4, pp. 123-134, 2013, [Online]. Available: https://doi.org/10.1145/2486001.2486020.

[3] K. Karpov, D. Kachan, V. Kirova, A. Ghermezian, D. Koshutina and E. Siemens, “Tunable multi-objective tree synthesis for application-layer multicast,” in Proceedings of the International Conference on Applied Innovations in IT (ICAIIT), Dec. 2025, pp. 68-73.

[4] N. Mareev, D. Kachan, K. Karpov, D. Syzov, E. Siemens and Y. Babich, “Efficiency of a PID-based congestion control for high-speed IP-networks,” in Proceedings of ICAIIT, vol. 6, no. 1, pp. 129-133, 2018, [Online]. Available: https://doi.org/10.13142/kt10006.45.

[5] M. Dong, Q. Li, D. Zats, J. R. Sokoll and I. Stoica, “PCC Vivace: Online-learning congestion control,” in Proceedings of the 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2018, pp. 343-356.

[6] S. Abbasloo, C.-Y. Yen and H. J. Chao, “Classic meets modern: A pragmatic learning-based congestion control for the Internet,” in Proceedings of the ACM SIGCOMM Conference, 2020, pp. 632-647, [Online]. Available: https://doi.org/10.1145/3387514.3405892.

[7] C. Liao, Y. Zhang and K. Winstein, “Astraea: Towards fair and efficient learning-based congestion control,” in Proceedings of EuroSys, 2024, [Online]. Available: https://doi.org/10.48550/arXiv.2403.01798.

[8] R. Galliera, A. Morelli, R. Fronteddu and N. Suri, “MARLIN: Soft actor-critic based reinforcement learning for congestion control in real networks,” in Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS), 2023, [Online]. Available: https://doi.org/10.1109/NOMS56928.2023.10154210.

[9] L. Giacomoni and G. Parisis, “Reinforcement learning-based congestion control: A systematic evaluation of fairness, efficiency and responsiveness,” in Proceedings of the IEEE INFOCOM Conference, 2024, pp. 1451-1460, [Online]. Available: https://doi.org/10.1109/INFOCOM52122.2024.10621288.

[10] F. Ruffy, M. Przystupa and I. Beschastnikh, “Iroko: A framework to prototype reinforcement learning for data center traffic control,” arXiv preprint arXiv:1812.09975, 2018, [Online]. Available: https://doi.org/10.48550/arXiv.1812.09975.