

PRISM: RL-Based Rate Control for Low Latency and High Data Rate in Mobile Edge Networks

Hamid Hassani, Staji Wilson, Kishor C. Joshi, Sonia M. Heemstra, Ignas G. Niemegeers, George Exarchakos
Eindhoven University of Technology, The Netherlands

{h.hassani,s.wilson,k.c.joshi,sheemstradegroot,i.g.m.niemegeers,g.exarchakos}@tue.nl

Abstract—Emerging delay-sensitive interactive services, such as extended reality and remote operation, demand advanced congestion control algorithms (CCAs) capable of maintaining high data rates while keeping latency low in mobile networks. Existing CCAs typically rely on congestion signals such as packet loss, delay, or explicit congestion notification (ECN) marks, which can be delayed, imprecise, or require extensive upgrades to the network infrastructure. To address these limitations, physical resource block (PRB) utilization has recently been proposed as a reliable early congestion signal in mobile networks [1]. This paper introduces PRISM, a reinforcement-learning-based CCA that uses peak PRB utilization to regulate the send rate under challenging mobility and fading conditions. NS3 simulation results show that, in a mobility scenario where two PRISM flows coexist with two TCP Cubic flows and one TCP BBR flow, PRISM maintains a per-packet mean one-way delay (OWD) of 6.3ms and a p_{99} OWD of 44.6ms, with negligible (0.02%) packet loss, over a 20min high-contention interval. However, over the same interval, the coexisting TCP flows experience $60\times$ higher mean OWDs of 371-390ms and p_{99} OWDs of 563-578ms, with 0.1% packet loss. PRISM achieves a $31\times$ improvement in goodput-over-latency performance, indicating a favorable tradeoff for delay-sensitive applications.

Index Terms—Mobile networks, low latency, congestion control, reinforcement learning, NS3 Simulation

I. INTRODUCTION

Enabling high data rate and low latency communication for next-generation interactive applications remains a key challenge for end-to-end congestion control algorithms (CCAs) in 5G and beyond mobile networks. Existing CCAs may prove insufficient for emerging applications that demand both high data rate and low delay [2].

Using packet loss as a congestion signal, as in TCP Cubic [3], often leads to significant delays, because the send rate is reduced only after substantial queue buildup has already occurred [4]. The use of round-trip time (RTT) or one-way delay (OWD) as congestion signals is also challenging: RTT (as in TCP BBR [5]) is known to be an imprecise indicator of congestion [6], while OWD is difficult to measure accurately in real-world deployments [7]. Furthermore, explicit congestion notification (ECN) mechanisms, such as L4S [8], are often impractical due to deployment and infrastructure upgrade difficulties.

Learning-based CCAs have been proposed to enhance adaptability under changing network conditions. However, recent results in [9] show that such CCAs, including PCC [10] and PCC-Vivace [11], can perform inconsistently in mobile networks, where mobility can cause rapid bandwidth

fluctuations that are difficult to infer from end-to-end transport signals. Furthermore, reinforcement learning (RL)-based CCAs, such as Orca [12] and ReCoCo [13], use RL to adapt congestion-control policies to complex network dynamics. However, they mainly depend on generic transport-layer signals, such as throughput, delay, and packet loss, which may be insufficient to achieve both high data rate and low latency [2].

A recent study showed that physical resource block (PRB) utilization can serve as an effective early indicator of congestion in mobile networks, as it is correlated with queueing delay at the RLC layer of the base station (BS) [1]. The proposed CCA, RAPID, regulates the send rate to maintain PRB utilization near a desired target, enabling both low latency and high data rate in private 5G edge networks. RAPID relies on a proportional-integral (PI) control policy with a fixed feedback gain, see Algorithm 1 in [1]. When wireless channel conditions vary due to, e.g., mobility and fading, this constant gain may become either overly aggressive or overly conservative, leading to unstable rate control and significant OWD spikes. Moreover, using only the average PRB utilization for rate control can hide short radio-resource bursts within a control interval. Such bursts can briefly build up the radio link control (RLC) queue and cause high OWD spikes [14], [15] even when the average PRB utilization remains below the target.

To address the above challenges, we present *PRISM*, a novel user-space transport solution based on proximal policy optimization (PPO) [16], in which the clipped objective stabilizes policy updates under noisy measurements and non-stationary contention. PRISM enables low delay, high data rate communication in challenging mobile network scenarios with user equipment (UE) mobility. It leverages cross-layer state features, including PRB utilization statistics and radio-quality indicators, to adjust the feedback gain at runtime. We implement PRISM in NS3 and evaluate its performance in the presence of UE mobility and cross-traffic.

The main contributions of this paper are as follows:

- We design and implement PRISM, a PPO-based CCA that uses cross-layer state features, including PRB utilization statistics and channel-quality information, to adapt its control gain and regulate the send rate based on peak PRB utilization in dynamic mobile environments.
- NS3 simulation results under UE mobility and TCP cross-traffic show that, in a scenario where two PRISM flows coexist with two Cubic flows and one BBR flow, PRISM maintains a mean OWD of 6.3ms and a p_{99}

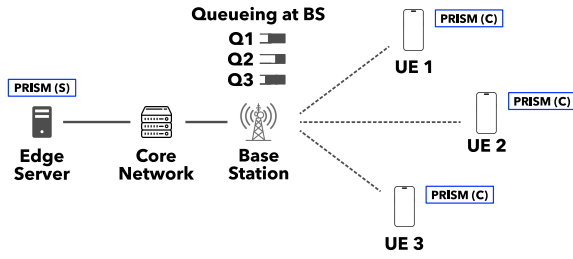


Figure 1. Edge transport architecture with the path bottleneck at the BS.

OWD of 44.6ms, about $60\times$ and $13\times$ lower, respectively, than the coexisting TCP flows, while achieving a $31\times$ improvement in goodput-over-latency performance.

The remainder of the paper is organized as follows. We present the system model in Section II. In Section III, we describe the design of PRISM, including its learning formulation and the training and model-selection procedure. Next, we discuss the performance evaluation results in Section IV and conclude the paper in Section V.

II. SYSTEM MODEL

We consider a 5G edge architecture, as depicted in Fig. 1, where the path bottleneck lies at the RLC layer of the BS [17]. The BS adheres to a per-data-radio-bearer (DRB) queueing policy [18]. The network is a private 5G in which all traffic flows are under the administration of the edge server (ES). We use the proportional-fair (PF) scheduler, which is widely adopted as a reference baseline in mobile network research [19]–[22]. The downlink traffic type is considered to be identical, e.g., best effort, so that all traffic flows towards a UE are carried by a single DRB, thus sharing the same RLC buffer at the BS. Moreover, we presume a full-buffer model, where there is always packet(s) queued in the RLC buffer of each UE at scheduling decisions [19].

III. PRISM DESIGN

We introduce a feedback loop that adjusts the downlink send rate at the ES. UE i collects cross-layer measurements, such as PRB utilization and transport-block (TB) retransmission ratio, and reports this information to the ES via UDP every δ seconds. The ES operates over control intervals of duration Δ ($> \delta$) and uses the received reports to construct the PPO state and update the send rate of UE i .

RAPID [1] updates the send rate of UE i as follows:

$$r_{i,k+1}^{\text{RAPID}} \leftarrow r_{i,k}^{\text{RAPID}} - K \cdot \gamma \cdot (\mu_{\text{PRB}_i}(\Delta_k) - P_{\epsilon,i}) \quad (1)$$

where $r_{i,k}^{\text{RAPID}}$ is the send rate of UE i , $\mu_{\text{PRB}_i}(\Delta_k)$ denotes its average PRB utilization over Δ_k , and $P_{\epsilon,i}$ is its target PRB utilization. The parameter $K > 0$ is a constant control gain that balances the convergence rate and stability of RAPID, and γ is a scaling factor that converts the PRB utilization error into a rate adjustment. Typical design parameter values are $\Delta = 200\text{ms}$, $\delta = 50\text{ms}$, and $K = 0.5$ [1].

Through NS3 simulations, we observed that using average PRB utilization, as in RAPID, can mask short bursts of high PRB utilization that arise under UE mobility, e.g., when UEs

Algorithm 1 PRISM feedback loop for adaptive rate control.

```

Initialize  $k \leftarrow 1$ 
while true do
  sleep ( $\Delta$ )
   $P_{\text{PRB},i}^{\max}(\Delta_k) \leftarrow \max_{j \in \mathcal{N}_{i,k}} \text{PRB}_{i,j}$ 
  Build state  $s_{i,k}$  from recent feedback observations
  Select gain  $a_{i,k} \leftarrow \pi_{\theta}(s_{i,k})$ 
  if  $P_{\text{PRB},i}^{\max}(\Delta_k) < P_{\epsilon,i}$  then
     $a_{i,k} \leftarrow a_{i,k}/n$ 
  end if
   $r_{i,k+1} \leftarrow r_{i,k} - a_{i,k} \cdot \gamma \cdot (P_{\text{PRB},i}^{\max}(\Delta_k) - P_{\epsilon,i})$ 
   $k \leftarrow k + 1$ 
end while

```

experience favorable channel conditions due to close proximity to the BS. These transient bursts are critical early indicators of congestion in mobile networks, as they may indicate queue buildup at the BS [14]. This motivated us to explore alternative PRB-utilization statistics that better capture these transient high PRB utilization phases.

While one possible solution would be to reduce both the feedback interval δ and the control interval Δ , this approach has important drawbacks: a smaller δ increases uplink feedback overhead, while a shorter Δ can lead to more abrupt and unstable send-rate updates. Therefore, both effects are undesirable for scalable, reliable, and stable operation.

PRISM preserves RAPID's framework but (i) replaces the constant feedback gain K with an adaptive gain, and (ii) uses maximum PRB utilization instead of average PRB utilization. The PPO agent learns to select this gain based on a state formed from a short history of recent observations.

Let $\mathcal{N}_{i,k}$ denote the set of feedback reports delivered by UE i during interval Δ_k . We denote $P_{\text{PRB},i}^{\max}(\Delta_k) = \max_{j \in \mathcal{N}_{i,k}} \text{PRB}_{i,j}$ as the maximum PRB utilization of UE i during interval Δ_k , where $\text{PRB}_{i,j}$ is the PRB utilization of UE i within the interval δ_j . If $P_{\text{PRB},i}^{\max}(\Delta_k)$ is less than its target value $P_{\epsilon,i}$, the ES increases the send rate r_i and decreases it if $P_{\text{PRB},i}^{\max}(\Delta_k) > P_{\epsilon,i}$.

Unlike RAPID, which uses a fixed gain K in (1), PRISM uses a PPO-selected adaptive gain $a_{i,k}$ to determine the magnitude of the rate update for UE i during the control interval Δ_k . For rate increases, the implementation scales $a_{i,k}$ by $1/n$, where n denotes the number of active downlink flows. Thus, ramp-up becomes more conservative under higher contention, while the PPO-selected gain is used for rate decreases. This feedback control logic is implemented using the pseudocode shown in Algorithm 1.

A. PPO Formulation

The PPO agent is implemented at the ES and selects an adaptive gain used to adjust the send rate every Δ seconds. Since a single feedback report only partially captures the queue and channel state, the PPO state of each UE is formed from the last eight feedback observations to approximate the Markov property [12].

Each observation contains the following normalized features: the PRB utilization (over δ), minimum and maximum PRB utilization (over Δ), target PRB utilization, the TB retransmission ratio, send rate, modulation and coding scheme (MCS), and signal-to-interference-plus-noise ratio (SINR). MCS and SINR are complementary rather than redundant: MCS captures the realized link-adaptation outcome and saturates at high channel quality, whereas SINR reflects the instantaneous channel margin. The resulting state size is 64.

The PPO actor uses this state to select a discrete gain $a_{i,k}$ from the action set $\mathcal{A} = \{0.05, 0.1, 0.2, 0.3, \dots, 1.0\}$. PRISM uses two-layer multilayer perceptrons for the actor and critic, each with hidden layers of sizes [128, 64].

For policy training, the reward is computed from measurements observed after each rate update. It combines two objectives: (i) a tracking reward $\mathcal{R}_{\text{track}}$ that encourages the peak PRB utilization to remain close to the target PRB utilization, and (ii) a gain-selection reward $\mathcal{R}_{\text{gain}}$ that guides the policy toward an appropriate gain under different network conditions.

To define the tracking component, we first compute the normalized error of UE i during the k th control interval Δ_k as follows:

$$e_{i,k} = \frac{P_{\text{PRB},i}^{\max}(\Delta_k) - P_{\epsilon,i}}{P_{\epsilon,i}} \quad (2)$$

where $e_{i,k} > 0$ indicates PRB overutilization and $e_{i,k} < 0$ indicates underutilization. The tracking reward is defined as:

$$\mathcal{R}_{\text{track},i,k} = \exp^{-1.0 [e_{i,k}]^-} - \exp^{2.0 [e_{i,k}]^+} + 1 \quad (3)$$

where $[x]^+ = \max(x, 0)$ and $[x]^- = \max(-x, 0)$. This asymmetric form penalizes overutilization more strongly than underutilization to prevent excessive queue buildup at the BS.

$\mathcal{R}_{\text{track}}$ alone is insufficient for learning gain selection. When the error is small, which usually occurs under good tracking, different gain values produce similar rate updates, so an arbitrary gain within a reasonable range may perform similarly. However, gain selection is crucial during high-error bursts, where the tracking reward alone provides little guidance for selecting an appropriate gain. We therefore add a gain-selection reward that favors actions close to a target gain $a_{i,k}^*$:

$$\mathcal{R}_{\text{gain},i,k} = - \left| \frac{a_{i,k}}{\max(\mathcal{A})} - a_{i,k}^* \right| \quad (4)$$

The target gain $a_{i,k}^*$ follows a fast-decrease, slow-increase strategy:

$$a_{i,k}^* = \begin{cases} \eta_{i,k}/\eta_{\max} & e_{i,k} > 0 \\ \frac{1}{4} \cdot \min\left(\frac{|e_{i,k}|}{0.3}, 1\right) \cdot \eta_{i,k}/\eta_{\max} & e_{i,k} \leq 0 \end{cases} \quad (5)$$

where $\eta_{i,k}$ denotes the spectral efficiency of UE i during control interval Δ_k , and $\eta_{\max}$ is the maximum spectral efficiency used for normalization [23]. The factor $\eta_{i,k}/\eta_{\max}$ scales the gain with channel quality: as η increases, each PRB carries more data, so the same PRB error requires a larger rate correction. When peak PRB utilization exceeds the target, $\mathcal{R}_{\text{gain}}$ encourages a stronger gain, independent of the

Table I
PRISM CONTROL AND PPO HYPERPARAMETERS.

Parameter	Value	Parameter	Value
Feedback interval δ	50ms	Discount γ_{PPO}	0.95
Control interval Δ	200ms	GAE λ_{PPO}	0.95
RAPID baseline gain K	0.5	Clip ϵ	0.2
PRB target factor	0.90	Entropy coefficient	0.02
PRBs per slot	106	Learning rate	10^{-3}
Rate scaling γ	10^6	On-policy batch size	512
History length m	8	Mini-batch size	64
State size	64	PPO epochs	4
Hidden layers	[128, 64]	$ \mathcal{A} $	11

error magnitude, to sharply decrease the send rate. When peak PRB utilization is below the target, $\mathcal{R}_{\text{gain}}$ encourages a more gradual, error-proportional rate increase scaled by the current spectral efficiency.

To avoid large reward magnitudes and improve PPO training stability, we clip the total reward as follows:

$$\mathcal{R}_{i,k} = \text{clip}(\mathcal{R}_{\text{track},i,k} + \mathcal{R}_{\text{gain},i,k}, -2, 1) \quad (6)$$

where $\text{clip}(x, \ell, u)$ constrains x to the interval $[\ell, u]$. Since $\mathcal{R}_{\text{track}} \leq 1$ and $\mathcal{R}_{\text{gain}} \leq 0$, the upper bound preserves the natural maximum reward, while the lower bound prevents large tracking errors from producing excessively negative rewards that could dominate PPO updates. The reward parameters were empirically tuned to penalize overutilization more strongly while maintaining stable PPO training.

B. Training and Model Selection

PRISM performs a policy update every 512 samples during training, with each sample collected over one control interval Δ . It computes generalized advantage estimates (GAE) and performs 4 PPO optimization epochs over each on-policy batch using mini-batches of 64 samples. Thus, each on-policy batch is reused across 4 epochs, with each epoch consisting of 8 mini-batches, after which the samples are discarded and fresh on-policy experience is collected. Table I summarizes the control and PPO hyperparameters.

Training uses multiple 2,000sec runs, each with one PRISM flow beginning at the start of the simulation and two TCP flows introduced later: one TCP Cubic and one TCP BBR. The TCP start times are drawn uniformly from [300, 400]sec. Their stop times are also randomized, with the TCP Cubic flow remaining active for at least 1,500sec and the TCP BBR flow for at least 750sec, while both flows end no later than 100sec before the end of the simulation. Each training run samples a new set of timings, thereby exposing the policy to diverse contention evolution patterns. The policy is then evaluated on a fixed 180sec validation scenario with one PRISM flow running for the entire simulation duration and one TCP BBR flow active from 60sec to 120sec. Table II summarizes the training, validation, and test scenarios.

Model selection uses a validation score rather than the raw PPO reward. This is necessary because the reward focuses on PRB tracking and gain selection, while our evaluation

Table II
TRAINING, VALIDATION, AND TEST SCENARIOS.

Scenario	PRISM	TCP	TCP variants	Duration
Train	1	2	1 Cubic, 1 BBR	2,000sec
Validation	1	1	1 BBR	180sec
Test	2	3	2 Cubic, 1 BBR	2,000sec

Table III
VALIDATION PERFORMANCE FOR THE SELECTED PPO TRAINING RUN.

Iteration	Goodput (Mbps)	Mean OWD (ms)	p_{95} (ms)	Max OWD (ms)	Score
1	37.7	7.01	15.7	155.7	11.82
2	37.7	7.01	15.7	155.7	11.82
3	36.2	4.39	10.9	53.6	14.93
4	36.0	3.71	7.9	31.3	16.82
5	35.0	3.72	7.6	31.4	16.48
6	35.3	4.10	8.7	58.1	15.64
7	33.6	3.44	6.0	16.2	16.91
8	32.4	3.31	5.0	13.5	16.96
9	32.6	3.21	4.6	11.1	17.41
10	33.5	3.26	4.7	11.8	17.74

objective is the goodput-latency tradeoff. The validation score is computed over the 60sec overlap window and is defined as:

$$\text{score} = \frac{T_{\text{app}} \cdot (1 - P_{\text{loss}})}{1 + \text{owd}_{\text{mean}}/5\text{ms} + \text{owd}_{95}/20\text{ms}} \quad (7)$$

where T_{app} is the goodput, P_{loss} is the packet-loss ratio, owd_{mean} is the mean OWD, and owd_{95} is the 95th-percentile OWD over the validation overlap window. Note that OWD measurements are used only for model selection during validation, not as PPO observation features or as inputs during test.

We evaluate 40 training checkpoints and select the model with the highest validation score. Table III shows the first 10 checkpoints out of 40. The best checkpoint occurs at checkpoint 10: mean OWD is reduced to 3.26ms, p_{95} OWD to 4.7ms, and maximum OWD to 11.8ms, while the score reaches 17.74. Training continues to 40 checkpoints, but no later checkpoint improves this validation score, so checkpoint 10 is kept for testing.

IV. EVALUATION

We implement PRISM in the NS3 (v3.41) network simulator via the UDP protocol and use the 5G-LENA module [20] to simulate non-standalone 5G networks. Each UE collects cross-layer measurements and reports them to the ES every $\delta = 50\text{ms}$. The ES aggregates the received reports, constructs the PPO state, obtains the selected gain from the PPO agent, and updates the send rate every $\Delta = 200\text{ms}$. We use ns3-ai [24] to exchange measurements and actions between NS3 and the PPO agent via a shared-memory interface. Note that PRISM requires only 6B of additional payload per feedback report compared with RAPID, which reports only PRB utilization. The feedback size remains constant per UE and is independent of the application data rate.

Next, we configure the InH_OfficeOpen_LoS channel with fast fading in the n78 band, numerology 1, and 40MHz

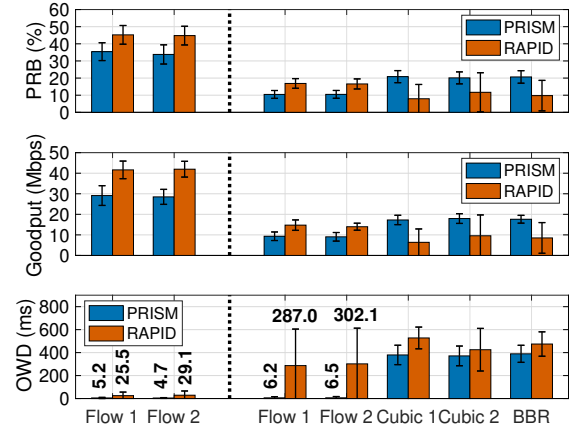


Figure 2. PRISM and RAPID performance under low and high contention during the test run. OWD values are computed on a per-packet basis. Error bars indicate one standard deviation.

bandwidth. We use the PF scheduler of the 5G-LENA module, i.e., `NrMacSchedulerOfdmaPF`. The traffic type is configured as eMBB, and the RLC layer operates in unacknowledged mode. The UEs follow the Gauss-Markov mobility model [25], with mean speed 1m/s, memory parameter $\alpha = 0.95$, and direction updates every 30sec, within a $200 \times 200\text{m}$ area centered on the BS. The observed MCS values are predominantly within the range 18-28.

For both PRISM and RAPID, the target PRB utilization is selected as $P_{e,i} = 0.9 \cdot \text{PRB}_{\text{max,slot}}/n$, where $\text{PRB}_{\text{max,slot}}$ is the maximum number of available PRBs per slot, determined by the bandwidth and the selected numerology, and n is the number of active downlink flows. The reduction factor of 0.9 keeps the target below the scheduler fair-share ceiling [1] to enhance system reliability and avoid OWD spikes caused by high PRB utilization.

A. Test Results

The test scenario includes two PRISM flows and three TCP cross-traffic flows, i.e., two TCP Cubic flows and one TCP BBR flow. The PRISM flows start at 1sec and 20sec and run until the end of the 2,000sec simulation. The TCP flows start at [100, 200, 300]sec and stop at [1600, 1700, 1800]sec, respectively. Therefore, the contention level changes over time, testing whether the learned gain can react to channel variation and cross-traffic changes. For example, peak contention occurs from $t = 300\text{sec}$, when the last TCP flow starts, to $t = 1600\text{sec}$, when the first TCP flow stops. During this period, the target PRB utilization remains at its lowest value, $P_e = 0.9 \cdot 106/5 \approx 19.1$, for more than 20min. RAPID is evaluated under the same mobility, traffic timing, and TCP coexistence setup used for PRISM, with $K = 0.5$ as specified in [1].

Fig. 2 compares the performance of PRISM and RAPID during two intervals of the test run: a low-contention interval in which only the two PRISM/RAPID flows, i.e., Flows 1 and 2, are active, and a high-contention interval in which these flows coexist with two TCP Cubic flows and one TCP BBR flow.

Table IV

TEST RESULTS OVER THE TEST RUN, WITH OWD STATISTICS COMPUTED FROM PER-FEEDBACK SAMPLES COLLECTED EVERY $\delta = 50\text{ms}$.

Scheme	Goodput (Mbps)	Mean OWD (ms)	p_{50} (ms)	p_{95} (ms)	p_{99} (ms)	Pckt loss (%)
PRISM	26.46	5.4	4.0	12.1	30.1	0.02
RAPID	39.35	254.9	15.8	703.9	761.5	21.2

Table V

TEST RESULTS ACROSS CONTENTION LEVELS FOR PRISM AND RAPID.

Scheme	$P_e(n)$	OWD (ms)					Goodput (Mbps)
		Mean	p_{50}	p_{95}	p_{99}	Max	
PRISM	95.4 (1)	3.1	2.3	5.6	7.3	8.0	74.1
PRISM	47.7 (2)	5.3	5.3	6.9	11.3	103.7	53.3
PRISM	31.8 (3)	4.0	3.3	5.8	11.2	121.6	32.6
PRISM	23.9 (4)	4.4	3.4	7.6	21.9	194.7	23.7
PRISM	19.1 (5)	5.8	4.0	14.9	33.6	169.8	17.0
RAPID	95.4 (1)	4.6	2.7	12.9	16.7	18.9	94.8
RAPID	47.7 (2)	27.5	10.1	114.2	160.2	253.4	77.2
RAPID	31.8 (3)	181.5	30.4	424.4	464.2	647.9	46.2
RAPID	23.9 (4)	236.3	42.3	550.7	600.0	699.8	34.9
RAPID	19.1 (5)	319.4	109.8	720.5	773.6	973.5	27.1

When only Flows 1 and 2 are active, PRISM maintains lower OWD for both flows, with mean OWDs of 5.2ms and 4.7ms, compared with 25.5ms and 29.1ms for RAPID, while using lower PRB utilization (34.6% vs. 45.0%). Here, $P_e = 47.7$ PRBs, which is 45% of the total 106 available PRBs.

Under TCP coexistence, PRISM achieves an even larger latency reduction, with mean OWDs of 6.2ms and 6.5ms compared with 287.0ms and 302.1ms for RAPID. This significant reduction is achieved with lower average PRB utilization, i.e., 10.5% for PRISM compared to 16.7% for RAPID, against a target of 18% ($P_e = 19.1$). Moreover, PRISM maintains negligible packet loss and about 9Mbps goodput per flow, whereas the coexisting Cubic and BBR flows achieve 17.3-18.0Mbps but experience substantially higher mean OWD of 371-390ms. However, under RAPID, the TCP flows are throttled to 6.4-9.6Mbps and suffer even higher OWD, ranging from 425ms to 528ms.

Table IV presents statistics for the PRISM/RAPID flows across the entire duration of the experiment. The results show that RAPID obtains higher goodput, but at the cost of severe queue buildup and packet loss. In contrast, PRISM backs off earlier when maximum PRB utilization exceeds the target PRB utilization, reducing mean OWD from 254.9ms to 5.4ms and packet loss from 21.2% to 0.02%. The tail-latency results show the same trend: PRISM keeps p_{99} OWD at 30.1ms, while RAPID reaches 761.5ms. This tradeoff is preferable for delay-sensitive interactive services.

We break down the same test run by contention regime in Table V. Each regime corresponds to a different target PRB utilization P_e , which changes as flows start and stop during the run. The OWD statistics in Table V are computed from the per-feedback samples, each representing the mean OWD over a $\delta = 50\text{ms}$ reporting interval.

The contention-regime breakdown further shows that PRISM keeps p_{99} OWD within tens of milliseconds across

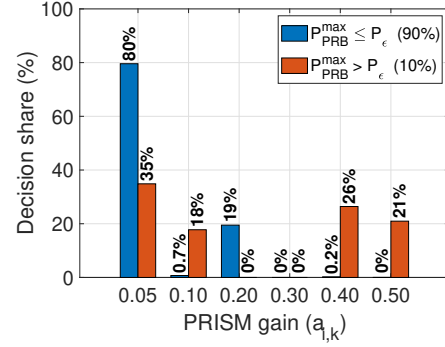


Figure 3. Distribution of the PPO-selected gain $a_{i,k}$ over the test run. The policy does not select gains in the range 0.6-1.0. NS3 simulation results.

all regimes, whereas RAPID degrades sharply as contention increases. This disparity is most pronounced at the lowest target PRB utilization, indicating that PRISM is most valuable under high contention. When $P_e = 19.1$, PRISM reduces mean OWD from 319.4ms to 5.8ms (55 $\times$ reduction) and reduces p_{99} OWD from 773.6ms to 33.6ms (23 $\times$ reduction). This latency improvement comes at the cost of a 37.3% decrease in goodput (37.4% reduction in PRB utilization) compared with RAPID, indicating that PRISM trades bandwidth utilization for lower delay by backing off more aggressively to prevent excessive RLC queue buildup.

To verify that RAPID's degradation is not primarily caused by the choice of its fixed feedback gain, we repeat the 2,000sec test for $K \in \{0.1, 0.25, 0.5, 1, 2, 5\}$. Across the tested values, RAPID's mean OWD remains between 255 and 497ms, p_{99} OWD between 760 and 787ms, and packet loss between 21% and 36%. Notably, $K = 0.5$ achieves the highest goodput and the lowest mean OWD and packet loss. For all tested gain values, average PRB utilization remains 7-14% below P_e , while peak PRB utilization exceeds P_e by 17-20%, indicating that tuning K alone does not eliminate the high-delay behavior of RAPID.

B. PPO Gain Analysis

We analyze the PPO-selected gain $a_{i,k}$ to better understand how PRISM reduces tail latency. We aggregate the control decisions of the two PRISM flows over the full test run, since both use the same policy. The selected gains are concentrated near the lower end of the action space $\mathcal{A}$: gain 0.05 accounts for about 75% of all decisions and gain 0.2 for about 18%, while larger gains in the range 0.4-0.5 appear in only about 5% of cases.

Fig. 3 shows this behavior more clearly by splitting the selected gains based on whether the peak PRB utilization falls below or exceeds the target PRB utilization P_e . When $P_{PRB}^{max} \leq P_e$, which represents roughly 90% of the total decisions, the policy selects the minimum gain of 0.05 in about 80% of the cases and only infrequently uses gains above 0.2. However, when overshoot occurs, the policy shifts toward higher gains, with 0.4 and 0.5 selected in approximately 47% of such instances. Thus, PRISM induces only small rate adjustments during typical operation but responds more aggressively when

Table VI
IMPACT OF THE TWO DESIGN COMPONENTS OF PRISM OVER THE
2,000SEC TEST RUN.

Scheme	PRB statistic	Feedback Gain	Goodput (Mbps)	Mean OWD (ms)	p_{95} (ms)	p_{99} (ms)	Pckt loss (%)
RAPID	Mean	0.5	39.35	254.9	703.9	761.5	21.2
Intermediate	Peak	0.5	34.34	15.7	46.8	405.0	0.23
PRISM	Peak	PPO	26.46	5.4	12.1	30.1	0.02

the peak PRB utilization surpasses the target PRB utilization and queue buildup is likely. This selective use of larger gains is consistent with the tail-latency reductions in Table IV.

C. Impact of the Design Components

The performance gains of PRISM result from complementary design choices: peak PRB utilization captures short resource bursts that may precede queue buildup, while the adaptive PPO gain controls the magnitude of rate adjustments under changing conditions. To assess the contribution of each component, we repeat the test run with an intermediate configuration that combines the peak statistic with the fixed gain $K = 0.5$. Table VI compares this configuration with RAPID and PRISM.

The intermediate configuration substantially lowers mean OWD from 254.9ms to 15.7ms and packet loss from 21.2% to 0.23%, with a 12.7% reduction in goodput. However, p_{99} OWD remains at 405.0ms. With the PPO-selected adaptive gain, PRISM further reduces mean OWD to 5.4ms and p_{99} OWD to 30.1ms, at an additional 22.9% reduction in goodput. These results show that the peak PRB statistic primarily reduces mean OWD and packet loss, while the adaptive gain provides additional control over tail latency.

D. Robustness Across Test Seeds

To assess robustness, we evaluate the PRISM policy over 10 independent test seeds using the entire 2,000sec test scenario, varying the fading and mobility realizations while keeping the traffic timing fixed. Across the 10 seeds, the median p_{99} OWD is 27.5ms for PRISM 1 and 30.7ms for PRISM 2, with corresponding standard deviations of 12.1ms and 11.4ms. The observed p_{99} OWD ranges from 10.8-51.8ms across both PRISM flows. These results indicate that PRISM maintains low tail latency despite variation in wireless conditions.

V. CONCLUSION

This paper introduced PRISM, a PPO-based CCA for mobile networks. PRISM leverages cross-layer metrics, including PRB utilization statistics, MCS, and TB retransmission ratio, to regulate the send rate. The NS3 simulation results show that PRISM significantly reduces OWD under mobility and TCP cross-traffic, achieving a favorable tradeoff between latency and goodput for delay-sensitive applications. The multi-seed evaluation further demonstrates that PRISM maintains low tail latency across varying fading and mobility settings, highlighting it as a promising low-latency CCA for mobile networks. We plan to benchmark PRISM against other RL-based CCAs and analyze its sensitivity to design parameters, as well as its robustness, fairness, and scalability. Moreover, a key objective

of our future work is to implement PRISM in a 5G testbed and evaluate its performance under real radio conditions, including its decision-making latency and computational overhead for real-time applications. Future work may explore distinct CCAs for different application classes, e.g., delay-critical and bandwidth-oriented traffic, by varying the PRB statistic used for rate control and the reward weighting.

ACKNOWLEDGMENTS

This work was supported by the HICONNECTS project under Grant Agreement No. 101097296 and by the Dutch National Growth Fund program Luchtvaart in Transitie (LiT).

REFERENCES

- [1] H. Hassani *et al.*, "RAPID: Proactive Rate Control for Enabling High Data Rate and Low Latency in 5G Edge Networks," in *Proc. INFOCOM Workshops*, 2025, pp. 1–6.
- [2] D. De Vleeschauwer *et al.*, "Application-Driven Traffic Differentiation," in *Proc. EuCNC/6G Summit*, 2024, pp. 1049–1054.
- [3] S. Ha *et al.*, "CUBIC: A New TCP-Friendly High-Speed TCP Variant," *SIGOPS Oper. Syst. Rev.*, vol. 42, no. 5, p. 64–74, 2008.
- [4] A. S. Tanenbaum, N. Feamster, and D. Wetherall, *Computer Networks*, 6th ed. Pearson, 2021.
- [5] N. Cardwell *et al.*, "BBR: Congestion-Based Congestion Control," *Commun. ACM*, vol. 60, no. 2, p. 58–66, 2017.
- [6] D. Chefrour, "One-Way Delay Measurement From Traditional Networks to SDN: A Survey," *ACM Comput. Surv.*, vol. 54, no. 7, 2021.
- [7] H. Hassani *et al.*, "Quick & Plenty: Achieving Low Delay & High Rate in 802.11ac Edge Networks," *Comput. Netw.*, vol. 187, p. 107820, 2021.
- [8] B. Briscoe *et al.*, "Low Latency, Low Loss, and Scalable Throughput (L4S) Internet Service: Architecture," RFC 9330, 2023.
- [9] J. Larrea *et al.*, "BISCAY: Practical Radio KPI Driven Congestion Control for Mobile Networks," in *Proc. NINEs*, 2026.
- [10] M. Dong *et al.*, "PCC: Re-Architecting Congestion Control for Consistent High Performance," in *Proc. NSDI*, 2015, p. 395–408.
- [11] —, "PCC Vivace: Online-Learning Congestion Control," in *Proc. NSDI*, 2018, p. 343–356.
- [12] S. Abbasloo *et al.*, "Classic Meets Modern: a Pragmatic Learning-Based Congestion Control for the Internet," in *Proc. SIGCOMM*, 2020, p. 632–647.
- [13] D. Markudova and M. Meo, "Advancing Congestion Control for Real-Time Communications With Reinforcement Learning: The ReCoCo Framework," *IEEE Trans. Netw. Serv. Manag.*, vol. 23, pp. 3998–4008, 2026.
- [14] A. Balasingam *et al.*, "Detecting if LTE is the Bottleneck with Burst-Tracker," in *Proc. MobiCom*, 2019.
- [15] A. Karnam *et al.*, "PINOCCHIO: Queueing-Onset Prediction in Mobile Edge Networks Using Reservoir Computing," in *Proc. ANRW*, 2026, pp. 24–26.
- [16] J. Schulman *et al.*, "Proximal Policy Optimization Algorithms," 2017.
- [17] H. Hassani *et al.*, "One-Way Delay Model for 5G Low Latency, High Data Rate Communications," in *Proc. ICC*, 2025, pp. 5859–5865.
- [18] M. Irazabal and N. Nikaein, "TC-RAN: A Programmable Traffic Control Service Model for 5G/6G SD-RAN," *IEEE J. Sel. Areas Commun.*, vol. 42, no. 2, pp. 406–419, 2024.
- [19] Y. Huang *et al.*, "GPF: A GPU-based Design to Achieve 100 μ s Scheduling for 5G NR," in *Proc. MobiCom*, 2018, p. 207–222.
- [20] N. Patriciello *et al.*, "An E2E Simulator for 5G NR Networks," *Simul. Model. Pract. Theory*, vol. 96, p. 101933, 2019.
- [21] OpenSim CTTC/CERCA, *NR Module User Manual*, 2024, release 3.3.
- [22] (2024) 5G MAC Scheduler Documentation. Accessed: May 10, 2026. [Online]. Available: <https://github.com/OPENAIRINTERFACE/openairinterface5g/blob/develop/doc/MAC/mac-usage.md>
- [23] ETSI, "5G; NR; Physical Layer Procedures for Data," ETSI, Technical Specification (TS) 138.214 v18.3.0, 2024.
- [24] H. Yin *et al.*, "ns3-ai: Fostering Artificial Intelligence Algorithms for Networking Research," in *Proc. WNS3*, 2020, p. 57–64.
- [25] T. Camp *et al.*, "A Survey of Mobility Models for Ad Hoc Network Research," *Wireless Commun. Mobile Comput.*, vol. 2, no. 5, pp. 483–502, 2002.