Jurnal Ilmu Komputer dan Informasi (Journal of Computer Science and Information)
16/2 (2023), 151-161. DOI: http://dx.doi.org/10.21609/jiki.v16i2.1159

Fine Tuning of Interval Configuration for Deep Reinforcement Learning Based Congestion Control

Haidlir Achmad Naqvi[1], Muhammad Hafizhuddin Hilman[1], Bayu Anggorojati[2]

[1] Faculty of Computer Science, Universitas Indonesia, Beji, Depok, 16424, Indonesia
[2] Monash University, Tangerang, 15345, Indonesia

Email: haidlir.achmad@ui.ac.id

Abstract

It is apparent that various internet services in today's digital ecosystem effectuate different types of networks' quality of services (QoS) requirements. This condition, in fact, adds another level of complexity to the current network congestion control protocols. Therefore, it drives the adoption of deep reinforcement learning to improve the protocols' adaptability to the dynamic networks' QoS requirements. In this case, the state-of-the-art works on congestion control protocols, formulate the markov decision process (MDP) by transforming the congestion control pattern from the saw tooth congestion window to the staircase sending rate per-interval cycles. This approach treats congestion control as a sequential decision-making process that fits reinforcement learning. However, the interval configuration parameter that gives the optimum QoS has not been empirically studied. In this work, we present an extensive study on various interval configuration parameters for the deep reinforcement learning-based congestion control agent. Our work shows that various interval configuration, which consists of the RTT estimator and the n parameter, results in different QoS. The experiment shows that the RTT$_{jk}$ has significantly higher throughput than RTT$_{ewma}$ and RTT$_{min-filtered}$ in various network conditions. Furthermore, we found that the RTT$_{jk}$ with $n = 2.0$ is superior to other configurations in almost all networking scenarios. Whereas the RTT$_{jk}$ with $n = 1.0$ is optimal for a network environment with fixed bandwidth scenario.

Keywords: *congestion control, deep reinforcement learning, interval duration*

1. Introduction

Modern computer networks, including the current Internet, 5G, and beyond, offer various services with diverse quality of services (QoS) requirements. Those QoS requirements add complexity to protocols and algorithms throughout all networking layers, including the congestion control within the transport layer. Congestion control is responsible for regulating the sending rate of end-to-end data transmission to avoid the Internet from congestion collapses [1]. The congestion shall occur if the load on the network is greater than the capacity of the network [2]. Empirically, congestion control is also able to optimize the QoS of youtube on five continents [3]. The traditional rule-based congestion control algorithms may work well in a certain scenario. However, the performance cannot be guaranteed in diverse network scenarios. Furthermore, the changing condition in one network scenario may also affect the algorithm's performance. Thus, intelligent congestion control is necessary [4]. The current state-of-the-art [5–11] leverages the power of reinforcement learning. Reinforcement learning is a variant of machine learning [12]. It improves the adaptability of the congestion control algorithm to face the challenge of modern computer networks.

The implementation of deep reinforcement learning to find the best policy behavior model requires the problem of congestion control to be formulated into the Markov decision process (MDP) as illustrated in fig 1. Recent works of congestion control based on deep reinforcement learning (DRL-CC) [5–8] use similar methods to formulate the MDP. They transform congestion control into an interval-based mechanism, as shown in fig 2. At each interval i,

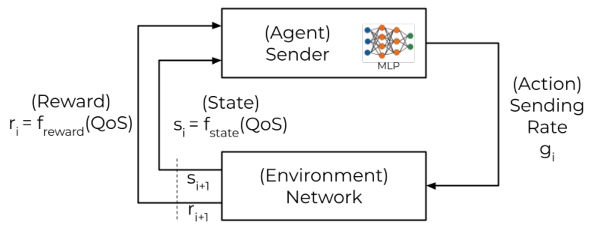
(Agent) Sender
MLP
(Action) Sending Rate g_i
(Reward) $r_i = f_{reward}(QoS)$
(State) $s_i = f_{state}(QoS)$
(Environment) Network
s_{i+1}
r_{i+1}

Figure 1. The MDP formulation of congestion control problem which uses multi-layered perceptron (MLP) to estimate the action.

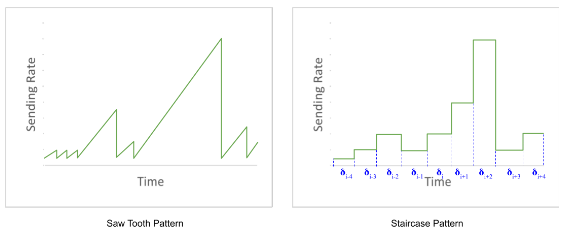

Figure 2. The MDP formulation transforms the congestion control sending rate pattern from the conventional saw tooth to staircase per-interval cycles.

this formulation uses a constant sending rate while continuously collecting and monitoring the network statistics to infer the QoS of the networks. The QoS of the interval i denoted as s_i later becomes the consideration to calculate sending rate at interval $i+1$, symbolized as g_{i+1}. In the training phase, the QoS also becomes the source of reward function r_i. In the MDP formulation of congestion control, the sending rate represents the actions, while the state and reward use QoS as the main source. From that formulation, one of the configurable variables is the interval duration δ_i. Aurora [5] defines interval as the product of n parameter and RTT$_{latest}$. Orca [6] and DeepCC [7] set their interval to a fixed value according to the prior knowledge of humans. NeuRoc [8] employs interval based on changepoint with constant maximum duration. Each method has a different interval definition. However, there is no empirical study for interval selection.

Our research follows the interval definition of Aurora [5], but the n parameter and RTT estimation are different. We pick several methods to estimate RTT inspired by decades of rule-based congestion control research [1, 3, 13, 14]. Our work shows the effect of interval duration based on those RTT estimators on the DRL-CC's network performance. The main goal is to find the appropriate value of n and RTT estimation used for DRL-CC. This research compares various RTT estimators and combines them with several multiplication factors n. This work examines each combination of the n parameter and RTT estimation in seven networking scenarios with various bandwidth, delay, and loss probability conditions. Our research uses statistical analysis [15] to analyze the data. Our findings are beneficial for the subsequent research and implementation of DRL-CC that uses interval-based monitoring. The key contributions of this paper are:

- To the best of our knowledge, this work is the first empirical study of the interval configuration, which includes the RTT estimator used in DRL-CC.
- It investigates the applicability of RTT$_{ewma}$ [13], RTT$_{jk}$ [1], and RTT$_{min-filtered}$ [14] as the RTT estimator of interval configuration.
- This paper discovers the preferred value of the n parameter and RTT estimation in the interval configuration examined in seven different networking scenarios.

The remainder of the paper is organized as follows. The following section presents the related works. Then, this paper explains the research methodology. Following that, it evaluates the result from extensive experiments. In the end, it summarizes the paper, concludes the work, and discusses future works.

2. Related Works

Aurora [5] establishes the foundation for reinforcement learning applications to solve congestion control problems. That algorithm maps the congestion control problem into the Markov decision process using PCC architecture [16]. Aurora [5] uses $\delta_i = n \times$ RTT$_{latest}$ as the interval. That work also tests various n values to discover its best performance. The performance evaluation shows that Aurora [5] is comparable to BBR [3] and Cubic [17]. Nevertheless, Aurora's model is not generalized yet. Moreover, that approach requires high computational resources.

Orca [6] and DeepCC [7] fuse rule-based and deep reinforcement learning to solve congestion control problems. They combine the best of both worlds. They run both approaches concurrently and pick the best one as the final sending rate. Orca [6] and DeepCC [7] define interval as the monitoring time period (MTP) and set it to a fixed value. They use 20ms as their MTP. They also study its performance on various MTP.

Libra [9] and TCP-NeuRoc [8] work similarly to Orca [6] and DeepCC [7]. However, they initially ran

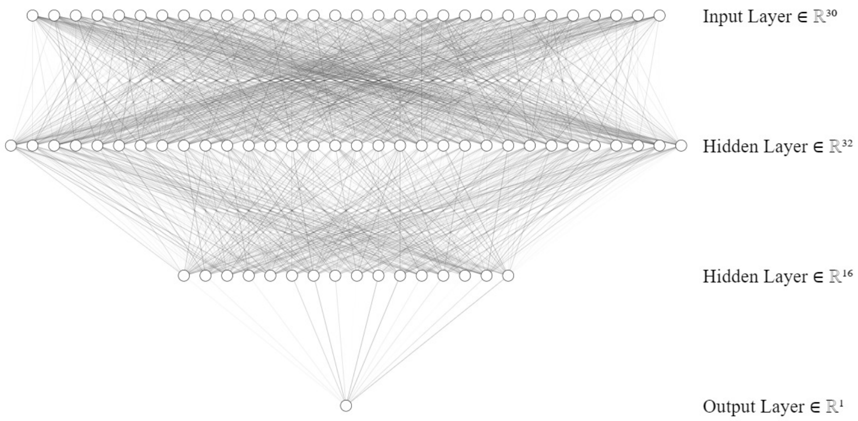
Input Layer ∈ $\mathbb{R}^{30}$
Hidden Layer ∈ $\mathbb{R}^{32}$
Hidden Layer ∈ $\mathbb{R}^{16}$
Output Layer ∈ $\mathbb{R}^{1}$

Figure 3. The policy model's architecture.

the rule-based congestion control to train the DRL agents using online training. Once they reach the maximum training duration, the agents can be used to estimate the sending rate decision. NeuRoc [8] exploits the DRL agent if a changepoint is detected. NeuRoc [8] employs variable length of monitoring interval (MI) based on abrupt variation from time series data, called changepoint, with a maximum duration of no more than $200ms$. On the other hand, Libra [9] utilizes the DRL agent interchangeably with the rule-based congestion control in each evaluation interval. Libra [9] uses evaluation interval (EI) to $0.5 \times$ RTT. MI and EI refer to the same variable, interval δ_i, as described in the previous section.

An approach called Teacher-Student Learning [10] and SymbolicPCC [11] leverage the distillation method to reinforcement learning. Teacher-Student Learning [10] teaches the DRL agent, as students, to imitate BBR [3], which represents the teacher. Unfortunately, there is no information about the interval configuration used in this work. SymbolicPCC [11] distills the Aurora [5] model as a teacher into a more interpretable model as a student in the form of the decision tree. SymbolicPCC [11] produces two kinds of symbolic policy, including the unbranched and the branched symbolic policy. This method uses a genetic algorithm to look for the symbolic policy and K-Means Clustering to branch the symbolic policy. This approach uses identical interval duration to Aurora [5].

We can summarize that the state-of-the-art employs different ways to determine the interval δ_i. Several methods, such as Orca [6], DeepCC [7], NeuRoc [8] set δ_i to an explicit and fixed time value. On the other hand, Aurora [5], SymbolicPCC [11] use the product of $n = 1.5$ and RTT_{latest} to calculate the δ_i. Quite similar to Aurora [5], Libra [9] use $n = 0.5$. Those variations in interval definition add repetitive work for subsequent DRL-CC's work to find their method's best δ_i. This indicates an open problem. Therefore, our work focuses on the empirical study to investigate the effect of several interval configurations on the network performance of DRL-CC.

3. Methodology

3.1. DRL Problem Formulation

This work follows the original Aurora design [5] that formulates the congestion control problem as a partially observable Markov decision process (POMDP). Aurora [5] formulates congestion control as a sequential decision-making problem under the RL framework. The agent's design adopts the PCC architecture [16] that divides the time into interval series. At the beginning of each interval, the agent adjusts its sending rate g_i. In the MDP formulation, the actions become changes of the sending rate. At each interval, the agent collects a networking event log, such as the packet size and the time of packet sent/acked/lost. At the end of each interval, we use the networking logs to calculate the quality of service (QoS): throughput (tput), delay, and loss. The QoS values are used to calculate the network statistics: the latency inflation, latency ratio, and sending rate ratio. The latency inflation is the ratio between the current latency to the previous latency. The latency ratio is the normalized current ratio to the minimum latency. The sending rate ratio is the normalized current sending rate to the throughput. We symbolize the calculation of latency inflation,

Table 1. Measurement scenario specification.

Network Scenario	Mean Bandwidth	One-Way Delay	Loss Probability	Queue Size
Default Pantheon [18]	12Mbps	60ms	0.0%	64packets
Static Movement [7]	21Mbps	20ms	0.0%	100packets
Walking Movement [7]	16Mbps	20ms	0.0%	100packets
Taxi Commuting [7]	21Mbps	20ms	1.0%	100packets
Bus Commuting [7]	19.8Mbps	20ms	1.0%	100packets
Low BDP [19]	2.64Mbps	88ms	0.0%	40packets
High BDP [19]	5.65MBps	200ms	0.0%	40packets

latency ratio, and sending rate ratio as $f_{state}(QoS)$. In the MDP formulation, the states are histories of network statistics $s_i = f_{state}(QoS)$. The reward function also uses the QoS values to calculate the reward at the end of the interval $s_i = f_{reward}(QoS)$. We follow the linear reward function defined by [5] as in eq 1.

$$f_{reward} = 10 \times tput - 10^3 \times delay - 2 \times 10^3 \times loss \quad (1)$$

The agent has a policy model that is responsible for receiving the states and inferring the action. The policy model adopts multi-layered perceptron architecture [20] with two hidden layers. The first and second hidden layers have 32 and 16 nodes, as illustrated in fig 3. The input layer consists of 30 nodes to accommodate the last ten states, with three network statistics in each state: latency inflation, latency ratio, and sending rate ratio. The output layer has only one node. All node uses tanh (hyperbolic tangent function) as the activation function. In the training process, we use proximal policy optimization (PPO) [21] as the training algorithm. The value function model adopts identical architecture to the policy model. The hyperparameter value for the discount rate γ is 0.99. The actor collects 8192 timesteps per batch. Each optimization uses 2048 timesteps data for 10 epochs. The learning rate is constant at 10^{-4}.

3.2. Experiment

We use a quantitative experimental approach to investigate several groups of interval configurations and test them statistically. Each group is the combination of the product of the n parameter and RTT estimation as shown in eq 3.2. We compare three RTT functions which are RTT$_{ewma}$ [13], RTT$_{jk}$ [1], and RTT$_{min-filtered}$ [14]. The multiplication factor n is set to $n \in N = [0.5, 0.75, 1.0, 1.5, 2.0, 3.0]$. Hence, we got 18 combinations of n and RTT estimator as interval configuration.

The RTT$_{ewma}$ takes inspiration from the exponential weight moving average in statistics. It is a type of memory that combines present and past RTT data to calculate the estimated RTT as defined in eq 2. The RTT$_{jk}$ complements the RTT$_{ewma}$ by taking the pseudo variance into the calculation of estimated RTT as shown in eq 3. It ensures that the estimated RTT exceeds the fluctuating RTT. The RTT$_{min-filtered}$ takes a different approach than the other two since it takes the minimum of RTT for the last 10 seconds. It considers that if routing changes, the routing table propagation needs 10 seconds to affect the RTT empirically.

$$\text{RTT}_{ewma} = (1-\alpha) \cdot \text{RTT}_{ewma} + \alpha \cdot \text{RTT}_{latest} \quad (2)$$

$$\text{RTT}_{var} = \beta \cdot \text{RTT}_{ewma} + (1-\beta) \cdot |RTT_{ewma} - RTT_{latest}|$$

$$\text{RTT}_{jk} = \text{RTT}_{ewma} + 4 \times \text{RTT}_{var} \quad (3)$$

$$\delta_i = n \times RTT \quad (4)$$

, where $RTT = \{RTT_{ewma}, \text{RTT}_{jk}, \text{RTT}_{min-filtered}\}$

We use Pantheon [18] to evaluate the network performance of each method. Pantheon [18] is a system that measures the performance of many transport protocols and congestion control schemes across a diverse set of network paths, either physical or emulated. It measures parameters such as mean throughput, 95th-%ile delay, and mean loss. The evaluation stage examines each combination of interval configurations in seven networking scenarios. The Pantheon default scenario is a link with fixed 12 Mbps bandwidth. The other four scenarios use cellular environment settings getting modeled as static movement, walking, taxi, and bus commuting [7]. The other two are low bandwidth-delay product (BDP) and high BDP networks [19]. Table 1 shows the network specifications for each measurement scenario. The measurement is done in 60 seconds of data transmission and repeated 30 times for each method evaluated. The evaluation is done locally inside a virtual machine (VM) with dedicated 1 vCPU and 4 GB RAM. We use GCP to provide the VM used in the measurement activity.

Table 2. Interval configurations group name.

Group Name	N	RTT Estimator
A	0.5	RTT_{ewma}
B	0.75	RTT_{ewma}
C	1.0	RTT_{ewma}
D	1.5	RTT_{ewma}
E	2.0	RTT_{ewma}
F	3.0	RTT_{ewma}
G	0.5	RTT_{jk}
H	0.75	RTT_{jk}
I	1.0	RTT_{jk}
J	1.5	RTT_{jk}
K	2.0	RTT_{jk}
L	3.0	RTT_{jk}
M	0.5	$\text{RTT}_{min-filtered}$
N	0.75	$\text{RTT}_{min-filtered}$
O	1.0	$\text{RTT}_{min-filtered}$
P	1.5	$\text{RTT}_{min-filtered}$
Q	2.0	$\text{RTT}_{min-filtered}$
R	3.0	$\text{RTT}_{min-filtered}$

Table 3. Manova table reporting the results of a multivariate comparison evaluating differences in the number of networking performance observed for default Pantheon scenario.

	Value	p value
Wilks' lambda	0.000507	0.0
Pillai's trace	2.363265	0.0
Hotelling-Lawley trace	71.612767	0.0
Roy's greatest root	49.550905	0.0

This paragraph presents the fixed variable used in this research. We use an identical machine-learning model for all interval combinations to focus the evaluation on the implementation aspect. The machine learning model design has been explained in section 3.1. The RTT_{ewma} uses $\alpha = \frac{1}{8}$. The RTT_{var} uses $\beta = \frac{3}{4}$. The pacer implements token bucket algorithm [22] with respect to the selected sending rate g_i.

We use one-way Manova [23] to analyze and find the significant difference between interval configurations involved in this research. The congestion control research can leverage the statistical method such as Analysis of Variance (ANOVA) [15, 24] to analyze the measurement data [25]. Anova is an approach to determine the data group's differences by inspecting the similarity between them [24]. There are 18 interval configurations or groups as independent variables value as defined in table 2. The measurements collect three kinds of data or dependent variables: throughput, delay, and loss. There are one independent variable and three dependent variables.

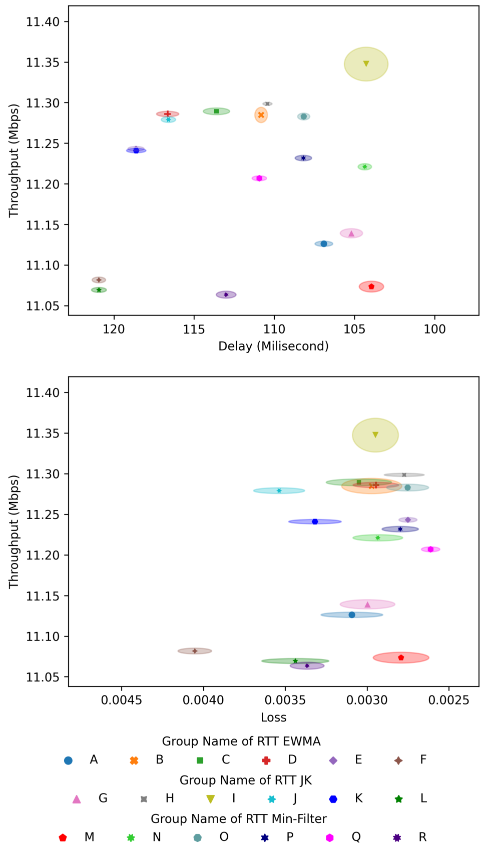

Figure 4. Throughput, delay, loss measurements at default Pantheon scenario [18].

4. Result

This section presents the performance measurement of each networking scenario in two kinds of scatter plots: (1) throughput-delay and (2) throughput-loss. We organize the diagrams in such a way that the extreme conditions are easy to identify. The top right area indicates high throughput and low delay or loss. It is the ideal position. On the contrary, the bottom left region denotes the low throughput and high delay or loss. Other quadrants show that there is a tradeoff between throughput and loss or delay.

We also use statistical analysis to test the significant relationship between each interval configuration. That value indicates whether there is a statistically significant difference in networking performance based on interval configuration in each networking scenario. Manova [23] calculates four multivariate test statistics: Wilks' Lambda, Pillai's Trace, Hotelling-Lawley Trace, and Roy's Greatest Root. Those four tests share the same null hypoth-

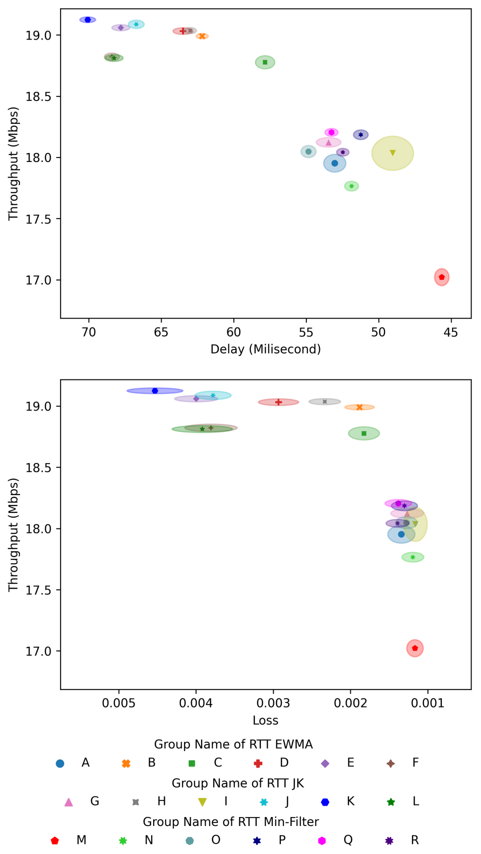

Figure 5. Throughput, delay, loss measurements at the cellular network with static position scenario [7].

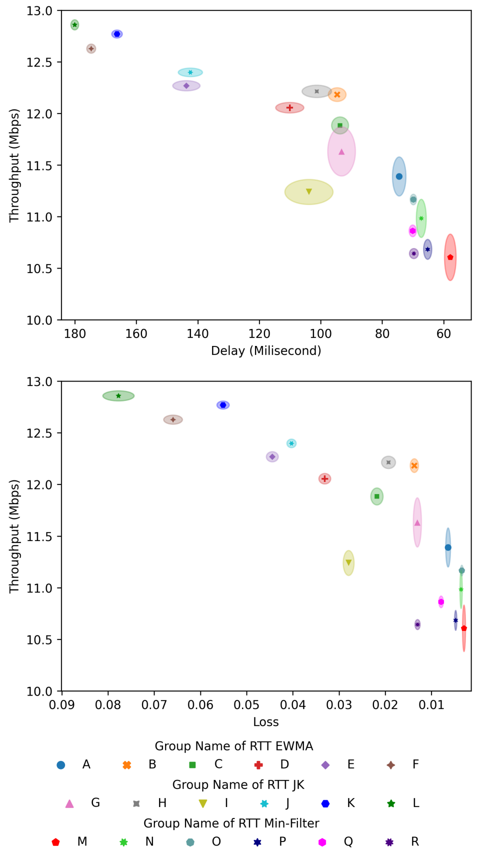

Figure 6. Throughput, delay, loss measurements at the cellular network with walking movement scenario [7].

Table 4. Manova table reporting the results of a multivariate comparison evaluating differences in the number of networking performance observed for static position scenario.

	Value	p value
Wilks' lambda	0.001696	0.0
Pillai's trace	2.225667	0.0
Hotelling-Lawley trace	50.354655	0.0
Roy's greatest root	42.073656	0.0

Table 5. Manova table reporting the results of a multivariate comparison evaluating differences in the number of networking performance observed for walking movement scenario.

	Value	p value
Wilks' lambda	0.001528	0.0
Pillai's trace	2.164291	0.0
Hotelling-Lawley trace	93.779338	0.0
Roy's greatest root	89.839293	0.0

esis. The value column, as shown in table 3-9, presents the calculated score of each multivariate test statistic. Later, those values are used to calculate the p-value. The null hypothesis is evaluated with regard to this p-value. We reject the null hypothesis that the group has no effect when the p-values are all less than .05.

We present the measurement result of the default Pantheon scenario in figure 4. Group I ($1.0 \times \text{RTT}_{jk}$) reaches the highest throughput among others. Group M ($0.5 \times \text{RTT}_{min-filtered}$) preserves low delay and low loss by sacrificing the throughput. Group I ($1.0 \times \text{RTT}_{jk}$) also balances the throughput, delay, and loss. Table 3 shows that all multivariate measures agree that there is a significant difference (p value < 0.05) in the measurement data. Those results deduce that Group I ($1.0 \times \text{RTT}_{jk}$) is superior in this default pantheon scenario.

Figure 5 plots the measurement result of the static position scenario. Group M ($0.5 \times \text{RTT}_{min-filtered}$) keeps the delay and loss low by dropping the throughput. Group K ($2.0 \times \text{RTT}_{jk}$)

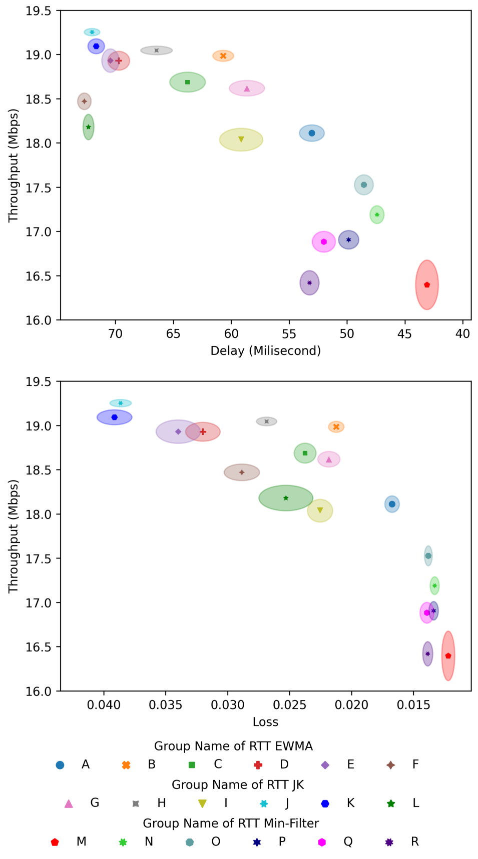

Figure 7. Throughput, delay, loss measurements at the cellular network in taxi commuting scenario [7].

Table 6. Manova table reporting the results of a multivariate comparison evaluating differences in the number of networking performance observed for taxi commuting scenario.

	Value	p value
Wilks' lambda	0.001396	0.0
Pillai's trace	2.541112	0.0
Hotelling-Lawley trace	36.842862	0.0
Roy's greatest root	28.534529	0.0

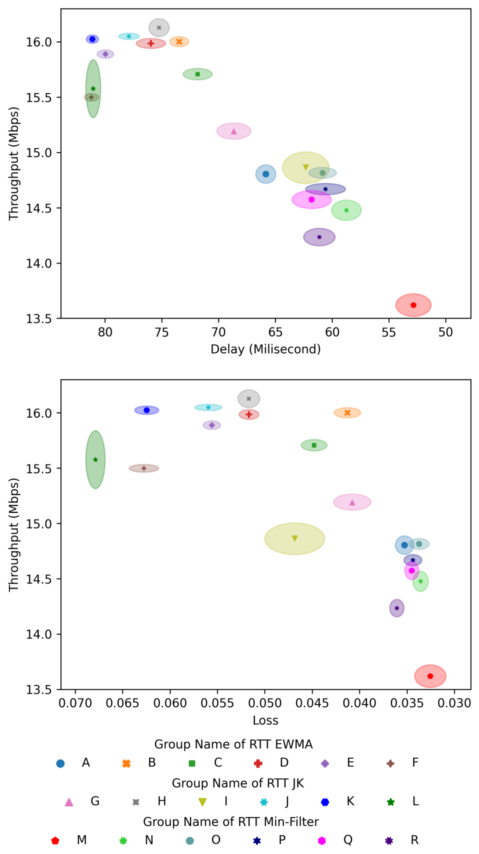

Figure 8. Throughput, delay, loss measurements at the cellular network in bus commuting scenario [7].

Table 7. Manova table reporting the results of a multivariate comparison evaluating differences in the number of networking performance observed for bus commuting scenario.

	Value	p value
Wilks' lambda	0.003224	0.0
Pillai's trace	2.208646	0.0
Hotelling-Lawley trace	33.175822	0.0
Roy's greatest root	26.725973	0.0

lead the throughput performance. Group C ($1.0 \times$ RTT$_{ewma}$) stabilizes the throughput, delay, and loss. However, the throughput of Group C ($1.0 \times$ RTT$_{ewma}$) is only 5% less than the Group K ($2.0 \times$ RTT$_{jk}$). Moreover, the statistical analysis shows that there is a significant difference (p value < 0.05) in the measurement data, as shown in table 4. Those measurement outputs infer that both group C ($1.0 \times$ RTT$_{ewma}$) and K ($2.0 \times$ RTT$_{jk}$) are recommended to cellular with static position scenario.

This work draws the measurement result of each interval configuration for the walking movement scenario in figure 6. Group M ($0.5 \times$ RTT$_{min-filtered}$) suppresses the delay and loss by retaining the throughput. Group H ($0.5 \times$ RTT$_{jk}$), B ($0.75 \times$ RTT$_{ewma}$), and C ($1.0 \times$ RTT$_{ewma}$) harmonize the throughput, delay, and loss. Group L ($3.0 \times$ RTT$_{jk}$), F ($3.0 \times$ RTT$_{ewma}$), and K ($2.0 \times$ RTT$_{jk}$) become the top 3 in throughput measurement. The throughput margin from groups L, F, and K to groups H, B, and C is less than 8%, but the delay margin is more than 80%. All multivariate measures agree that

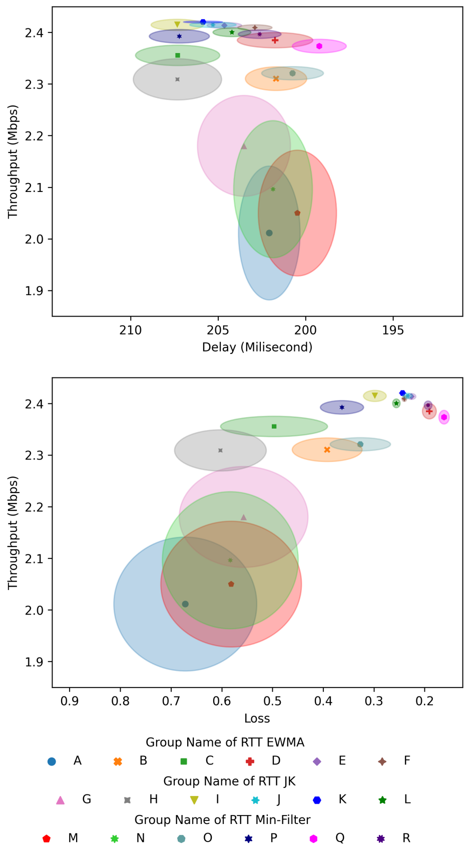

Figure 9. Throughput, delay, loss measurements at low bandwidth-delay product scenario.

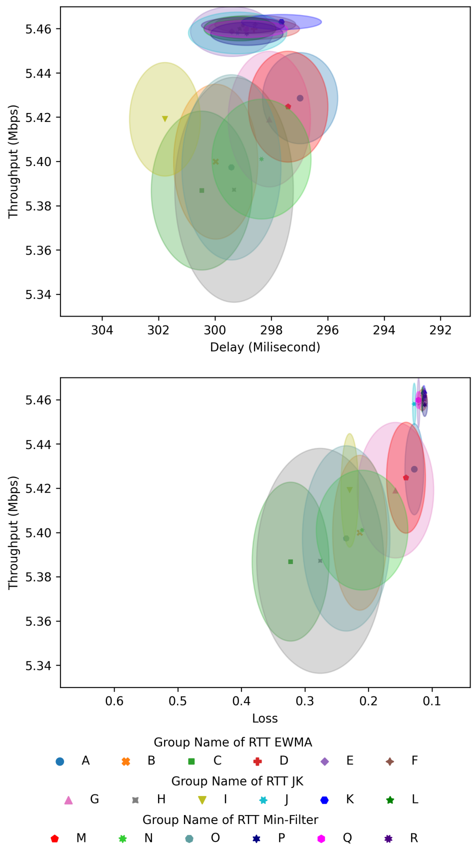

Figure 10. Throughput, delay, loss measurements at high bandwidth-delay product scenario.

Table 8. Manova table reporting the results of a multivariate comparison evaluating differences in the number of networking performance observed for low bandwidth-delay product scenario.

	Value	p value
Wilks' lambda	0.153232	0.0
Pillai's trace	1.230026	0.0
Hotelling-Lawley trace	3.262024	0.0
Roy's greatest root	2.484698	0.0

Table 9. Manova table reporting the results of a multivariate comparison evaluating differences in the number of networking performance observed for high bandwidth-delay product scenario.

	Value	p value
Wilks' lambda	0.516508	0.0
Pillai's trace	0.548742	0.0
Hotelling-Lawley trace	0.81294	0.0
Roy's greatest root	0.630792	0.0

there is a significant difference (p value < 0.05) in the measurement data, as shown in table 5. Those measurement results reveal that groups H, B, and C are more favored for delay-sensitive applications in this scenario.

We show the measurement result of each interval configuration for the taxi commuting scenario in figure 7. Group J ($1.5 \times \text{RTT}_{jk}$) and K ($2.0 \times \text{RTT}_{jk}$) lead throughput measurement respectively. Group A ($0.5 \times \text{RTT}_{ewma}$) balances the throughput, delay, and loss. Group M ($0.5 \times \text{RTT}_{min-filtered}$) suppresses the delay and loss by keeping the throughput low. Group A's throughput is only less than 5% of group J and K. Whereas group A's delay and loss are almost half of the group J and K. Table 6 shows that the multivariate measures agree that there is a significant difference (p value < 0.05) in the measurement data. Those results deduce that Group A ($0.5 \times \text{RTT}_{ewma}$) is preferable to this scenario.

Our work draws figure 8 to plot the measurement result of each interval configuration for the bus commuting scenario. Group B ($0.75 \times \text{RTT}_{ewma}$)

Table 10. Summary table of each evaluation scenario.

Scenario	Higher Throughput	Lower Delay and Loss
Default Pantheon	I	M
Static Position	C and K	C and M
Walking Movement	F, L and K	C and M
Taxi Commuting	J and K	A and M
Bus Commuting	H, J, and K	B, C, and M
Low BDP	J and K	M and Q
High BDP	K and Q	A, K, M

and C ($1.0 \times$ RTT$_{ewma}$) harmonize the throughput, delay, and loss. Group H ($0.5 \times$ RTT$_{jk}$), J ($1.5 \times$ RTT$_{jk}$), and K ($2.0 \times$ RTT$_{jk}$) become the top 3 in throughput measurement. Group M ($0.5 \times$ RTT$_{min-filtered}$) suppresses the delay and loss by retaining the throughput. All multivariate measures agree that there is a significant difference (p value < 0.05) in the measurement data, as shown in table 7. Those measurement outputs infer that both group B ($0.75 \times$ RTT$_{ewma}$) and C ($1.0 \times$ RTT$_{ewma}$) are preferred to cellular with bus commuting scenario since the throughput is comparable to the highest throughput, but they have better delay and loss.

Figure 9 portrays the measurement result of each interval configuration for a low bandwidth-delay product scenario. Group Q ($2.0 \times$ RTT$_{min-filtered}$) stabilizes the throughput, delay, and loss. Group Q ($2.0 \times$ RTT$_{min-filtered}$) and M ($0.5 \times$ RTT$_{min-filtered}$) preserve the delay and loss by suppressing the throughput. Group K ($2.0 \times$ RTT$_{jk}$) and J ($1.5 \times$ RTT$_{jk}$) are the first and second highest throughput measurement. The distance of groups Q and M to groups K and J with respect to throughput, delay, and loss is small. Table 8 shows the multivariate measures agree that there is a significant difference (p value < 0.05) in the measurement data. Those results infer that Group Q ($2.0 \times$ RTT$_{min-filtered}$) is advisable for a low bandwidth-delay product scenario.

We plot the measurement result of each interval configuration for a high bandwidth-delay product scenario in figure 10. Group Q ($2.0 \times$ RTT$_{min-filtered}$) and K ($2.0 \times$ RTT$_{jk}$) become the top 2 in throughput measurement. Group K ($2.0 \times$ RTT$_{jk}$), M ($0.5 \times$ RTT$_{min-filtered}$), and A ($0.5 \times$ RTT$_{ewma}$) keep the delay and loss low by retaining the throughput. Group K ($2.0 \times$ RTT$_{jk}$) and A ($0.5 \times$ RTT$_{ewma}$) balances the throughput, delay, and loss. Group K and Q's delay and loss are comparable to the lowest delay and loss. Table 9 presents the multivariate measures that agree that there is a significant difference (p value < 0.05) in the measurement data. Those measurement results reveal that groups K and Q are effective in this scenario.

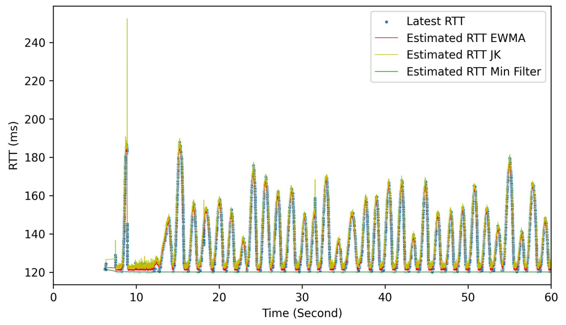

Figure 11. Sample RTT log of default Pantheon scenario.

Those measurement results in seven networking scenarios prove that certain networking condition prefers particular interval configuration for DRL-CC. Table 10 summarizes the evaluation results of each scenario. Group K ($2.0 \times$ RTT$_{jk}$) appears as the high throughput in six of seven scenarios, including a cellular network with fluctuating bandwidth. Whereas, group I ($1.0 \times$ RTT$_{jk}$) is better than group K ($2.0 \times$ RTT$_{jk}$) in the flatter bandwidth scenario. Group C ($1.0 \times$ RTT$_{ewma}$) reaches lower delay and loss but smaller throughput in three of seven scenarios. Group M ($0.5 \times$ RTT$_{min-filtered}$) always has a low delay and loss by retaining the throughput in all scenarios. The measurement of group M ($0.5 \times$ RTT$_{min-filtered}$) shows that narrow interval durations yield low delay and loss.

Those measurement results also show that RTT$_{jk}$ have higher throughput than RTT$_{ewma}$ and RTT$_{min-filtered}$. Fig 11 presents the recorded data of the latest RTT, estimated RTT$_{ewma}$, estimated RTT$_{jk}$, and estimated RTT$_{min-filtered}$. We can see that RTT$_{ewma}$ draws a line within the latest RTT data. Whereas RTT$_{jk}$ is consistently at the top of the latest RTT. While RTT$_{min-filtered}$ always becomes the lower bound. Those behavior affect the sending rate and, eventually, the throughput. This phenomenon relates to the selection of interval duration δ_i.

Intuitively, we should select interval duration δ_i that is greater than or equal to the propagation RTT or RTT_{prop}. RTT_{prop} is the minimum RTT of a communication. Let a packet is transmitted at time t; thus, the fastest ACK arrives at $t + RTT_{prop}$. If δ_i is less than RTT_{prop}, there are no networking statistics that can be collected where it is required by the policy model as input to infer the changes in sending rate. Therefore RTT_{prop} defines the lower bound of sampling interval δ_i. Since the network condition is stochastic over time, the interval δ_i should also be adaptive and follows the RTT trend. That intuition also explains the result in the default Pantheon scenario. The default Pantheon scenario has a fixed bandwidth; therefore, we need interval duration δ_i that is close to RTT_{prop}. Fig 11 shows that RTT$_{jk}$ is the closest among the other estimations. Hence, it explains why Group I ($1.0 \times$ RTT$_{jk}$) gets the highest throughput in default Pantheon scenario.

5. Conclusion

We present an empirical study of interval configuration on the DRL-CC agent's implementation. The use of interval configuration significantly produces different QoS measurement results. Our work also proves that RTT$_{jk}$ have higher throughput than RTT$_{ewma}$ and RTT$_{min-filtered}$ in various networking scenarios. Interval configuration, which combines RTT$_{jk}$ and $n = 2.0$, is recommended for almost all networking scenarios. Whereas interval configuration pairing RTT$_{jk}$ and $n = 1.0$ is optimal for a network with fixed bandwidth.

This work shows that there is a diverse interval configuration to balance the networking performance of the congestion control agents. There is a probability that static interval configuration unfits certain networking scenarios. Accordingly, we suggest the learnable interval configuration of DRL-CC for future work. The machine learning model can also estimate the interval duration δ_i, in addition to sending rate g_i, since multi-layered perceptron (MLP) is capable of inferring multiple outputs. Moreover, we are aware that the computational complexity of this DRL-CC prototype is high. It is caused by the use of UDT transport protocol. One of the potential solutions to address that issue is by replacing the transport protocol from UDT to QUIC. We plan to evaluate that method for future work.

References

[1] V. Jacobson, “Congestion avoidance and control,” *SIGCOMM Comput. Commun. Rev.*, vol. 18, no. 4, p. 314–329, aug 1988. [Online]. Available: https://doi.org/10.1145/52325.52356

[2] B. A. Forouzan, *Data communications and networking*. Huga Media, 2007.

[3] N. Cardwell, Y. Cheng, C. S. Gunn, S. H. Yeganeh, and V. Jacobson, “BBR: Congestion-Based Congestion Control,” *Queue*, vol. 14, no. 5, pp. 20–53, 2016.

[4] H. Jiang, Q. Li, Y. Jiang, G. Shen, R. Sinnott, C. Tian, and M. Xu, “When machine learning meets congestion control: A survey and comparison,” *Computer Networks*, vol. 192, p. 108033, 2021.

[5] N. Jay, N. Rotman, B. Godfrey, M. Schapira, and A. Tamar, “A deep reinforcement learning perspective on internet congestion control,” in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 3050–3059. [Online]. Available: https://proceedings.mlr.press/v97/jay19a.html

[6] S. Abbasloo, C.-Y. Yen, and H. J. Chao, “Classic meets modern: A pragmatic learning-based congestion control for the internet,” ser. SIGCOMM ’20. New York, NY, USA: Association for Computing Machinery, 2020, p. 632–647. [Online]. Available: https://doi.org/10.1145/3387514.3405892

[7] S. Abbasloo, C. Y. Yen, and H. J. Chao, “Wanna Make Your TCP Scheme Great for Cellular Networks? Let Machines Do It for You!” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 265–279, 2021.

[8] W. Li, S. Gao, X. Li, Y. Xu, and S. Lu, “TCP-NeuRoc: Neural Adaptive TCP Congestion Control with Online Changepoint Detection,” *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2461–2475, 2021.

[9] Z. Du, J. Zheng, H. Yu, L. Kong, and G. Chen, “A unified congestion control framework for diverse application preferences and network conditions,” in *Proceedings of the 17th International Conference on Emerging Networking EXperiments and Technologies*, ser. CoNEXT ’21. New York, NY, USA: Association for Computing Machinery, 2021, p. 282–296. [Online]. Available: https://doi.org/10.1145/3485983.3494840

[10] Y. Zheng, L. Lin, T. Zhang, H. Chen, Q. Duan, Y. Xu, and X. Wang, “Enabling Robust DRL-Driven Networking Systems via Teacher-Student Learning,” *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 376–392, 2022.

[11] S. P. Sharan, W. Zheng, K.-F. Hsu, J. Xing, A. Chen, and Z. Wang, “Symbolic distillation for learned TCP congestion control,” in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022. [Online]. Available: https://openreview.net/forum?id=rDT-n9xysO

[12] M. Klaiber and J. Klopfer, "A systematic literature review on SOTA machine learning-supported computer vision approaches to image enhancement," *Jurnal Ilmu Komputer dan Informasi*, vol. 15, no. 1, pp. 21–31, Feb. 2022. [Online]. Available: https://doi.org/10.21609/jiki.v15i1.1017

[13] M. N. ul Amin, "Memory type estimators of population mean using exponentially weighted moving averages for time scaled surveys," *Communications in Statistics - Theory and Methods*, vol. 50, no. 12, pp. 2747–2758, 2021. [Online]. Available: https://doi.org/10.1080/03610926.2019.1670850

[14] V. Arun and H. Balakrishnan, "Copa: Practical delay-based congestion control for the internet," in *Proceedings of the Applied Networking Research Workshop*, ser. ANRW '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 19. [Online]. Available: https://doi.org/10.1145/3232755.3232783

[15] H.-Y. Kim, "Analysis of variance (ANOVA) comparing means of more than two groups," *Restorative Dentistry & Endodontics*, vol. 39, no. 1, p. 74, 2014. [Online]. Available: https://doi.org/10.5395/rde.2014.39.1.74

[16] M. Dong, Q. Li, D. Zarchy, P. B. Godfrey, and M. Schapira, "Pcc: Re-architecting congestion control for consistent high performance," in *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation*, ser. NSDI'15. USA: USENIX Association, 2015, p. 395–408.

[17] S. Ha, I. Rhee, and L. Xu, "CUBIC," *ACM SIGOPS Operating Systems Review*, vol. 42, no. 5, pp. 64–74, jul 2008. [Online]. Available: https://dl.acm.org/doi/10.1145/1400097.1400105

[18] F. Y. Yan, J. Ma, G. D. Hill, D. Raghavan, R. S. Wahby, P. Levis, and K. Winstein, "Pantheon: The training ground for internet congestion-control research," in *Proceedings of the 2018 USENIX Conference on Usenix Annual Technical Conference*, ser. USENIX ATC '18. USA: USENIX Association, 2018, p. 731–743.

[19] V. Jacobson and R. Braden, "Tcp extensions for long-delay paths," RFC 1072, Internet Engineering Task Force, RFC 1072, Oct. 1988. [Online]. Available: https://www.rfc-editor.org/info/rfc1072

[20] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[21] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347

[22] M. Bosk, M. Gajic, S. Schwarzmann, S. Lange, and T. Zinner, "Htbqueue: A hierarchical token bucket implementation for the omnet++/inet framework," *CoRR*, vol. abs/2109.12879, 2021. [Online]. Available: https://arxiv.org/abs/2109.12879

[23] V. Todorov and P. Filzmoser, "Robust statistic for the one-way MANOVA," *Computational Statistics & Data Analysis*, vol. 54, no. 1, pp. 37–48, Jan. 2010. [Online]. Available: https://doi.org/10.1016/j.csda.2009.08.015

[24] A. S. Indrawanti and W. Wibisono, *Jurnal Ilmu Komputer dan Informasi*, vol. 8, no. 2, p. 92, Aug. 2015. [Online]. Available: https://doi.org/10.21609/jiki.v8i2.307

[25] H. Wang, J. Tang, and B. Hong, "Research of wireless congestion control algorithm based on EKF," *Symmetry*, vol. 12, no. 4, p. 646, Apr. 2020. [Online]. Available: https://doi.org/10.3390/sym12040646