Multi-Context-aware RL approach towards INT-based Congestion Control Algorithm

Ramyashree Venkatesh Bhat, Jetmir Haxhibeqiri, Ingrid Moerman, Jeroen Hoebeke
IDLab, Ghent University – imec, Ghent, Belgium
Email: [name.surname]@ugent.be

***Abstract*—Diverse applications in private networks need to be optimally served over both wired and wireless. One way to do so is by having more intelligent protocols, as current designs only have very limited feedback. This paper proposes a reinforcement learning-based congestion control algorithm for private wireless networks that takes into account application information and network context. Three algorithm designs are presented in the paper, and one was implemented and tested, showing adaptability to changing contexts.**

***Index Terms*—congestion control, context-aware reinforcement learning, INT, MDP**

I. Introduction

Private wireless networks are increasingly adopted across sectors like industry automation, healthcare, and military due to their enhanced coverage, performance, security, and adaptability to artificial intelligence (AI) and Internet of Things (IoT) technologies. These networks must meet the high demands of applications like AR/VR and cloud computing in terms of throughput, latency, reliability, and application prioritization. Though techniques like slicing, priority queuing, and differentiated services (DiffServ) are offered by the intermediate network nodes, the massive increase in traffic (number of flows in the range of 100s–1000s) with different priority requirements ultimately exceeds their capabilities on incorporating new functionalities. Transmission Control Protocol (TCP) is widely used as end-to-end transport protocol for its reliable, error-free data transfer and congestion control. However, most congestion control algorithms are designed for wired networks, overlooking wireless-specific issues like interference, mobility, and lossy mediums. As network traffic grows, ensuring congestion-free services alongside reliability and service differentiation is critical in wireless networks.

The transition to Software Defined Networking (SDN), which separates control and data planes, allows for full programmability of the forwarding plane [1]. To maximize this, understanding application requirements and real-time network context is crucial. In-band network telemetry (INT) techniques, superior to out-of-band methods, have been adapted for wireless networks [2]. An application-network interaction (APP-NET) interface designed in [3] allows applications to specify their traffic requirements to the network layer and understand the real-time network context. Design goals for the future private wireless networks (6G and Wi-Fi)[1] involve integration of INT and APP-NET in making decisions. The end devices connecting to private networks are also now more powerful in terms of memory, computation, and programmability than before. Innovation in Linux kernels like eBPF has enabled the programmability of network protocol stacks in the end devices [4].

Leveraging the computational capabilities of the end devices and data programmability of the forwarding plane along with INT and APP-NET integration, this paper proposes a solution where the end devices collaborate with the intermediate network nodes in achieving optimal network operation. As a first step, a novel multi-context-aware reinforcement learning-based (RL-based) congestion control (CC) algorithm is introduced that adapts the number of application data packets sent to the network based on application requirements and real-time network information. Since the wireless network parameters such as channel quality, other competing nodes, and number of flows, being independent of the actions taken by an end device, have a significant influence on the overall throughput and latency of the end device, the solutions proposed in this paper are based on multi-context-aware RL-based CC that is modeled as a contextual Markov Decision Process (CMDP).

The rest of this paper is structured as follows: Section II focuses on related works in the area of RL-based CC algorithms. Section III elaborates on the challenges and Section IV explains the proposed solutions for implementing an RL-based solution for wireless networks. Section V gives an overview of preliminary results using Q-learning. Section VI concludes the paper with insights on the future work.

II. Related work

This section discusses research works that focus on reinforcement learning-based congestion control algorithms. The survey paper [5] gives a detailed description of different RL techniques used to implement CC algorithms for different network scenarios. Some of the commonly used value-based techniques are Q-learning and deep Q-learning [6], [7], whereas policy-gradient, Actor-Critic (AC), Advantage Actor-Critic (A2C), and Asynchronous Advantage Actor-Critic (A3C) [8], [9] are commonly used value-based learning techniques. Based on the survey, most of the implementations update congestion window size ($cwnd$) for specific scenarios. A few of the algorithms also manage the queue length based on the current state obtained from congestion notification. In such scenarios, Proportional Integral Derivative (PID) is generally used in the

[1]https://www.bell-labs.com/research-innovation/projects-and-initiatives/unext/#gref

RL technique [10]. Despite showing strong learning capabilities, the current RL-based CC algorithms do not converge easily and find state abstraction challenging. In addition to that, the algorithms require a huge amount of storage for states and actions and have high computational complexity.

In [11], the authors have proposed an adaptive CC algorithm for wireless networks based on deep-RL, where the algorithm classifies the flow using deep neural networks and finds the appropriate $cwnd$ using a global repository based on the flow classification. The solution is implemented and tested in a simulator. The authors of [12] have proposed TCP-Gvegas, an improved TCP Vegas based on grey prediction theory for multi-hop ad hoc networks. The algorithm uses an optimal exploration method based on Q-learning and round-trip time quantizer.

III. Challenges

One of the main challenges is the ever-changing dynamics and unpredictability of wireless networks. Due to the presence of several dynamic network parameters, like channel conditions, intermediate network node context, and competing nodes in private wireless networks, it is challenging to confine the state and action space. With the addition of diverse requirements for applications, the state and action space increases exponentially. In addition to that, it is essential to achieve short-term performance enhancement (such as throughput, latency) along with the long-term goal (congestion-free, service differentiation). Since service differentiation and adaptability are the core features of private networks, it is crucial to address these challenges. The challenges involved in designing an RL-based solution specific to Q-learning-based algorithms for wireless networks are outlined in [13]. Some of the important challenges involve adaptation of Q-table to abruptly changing environment, effects of exploration on the stability of the algorithm, and achieving stable Q-values. The other issues addressed are in the area of multi-agent RL algorithms.

IV. Proposed solution

A. RL model

The existing CC algorithms decide the $cwnd$ based on the partial information available from acknowledgment (ACK) packets. Keeping in mind the goals of future private wireless networks, it is essential to integrate INT and APP-NET. Leveraging this, an RL-based CC algorithm that utilizes real-time network information from INT is designed. Some of the INT monitoring parameters that are relevant for CC algorithms are buffer capacity, number of packets arriving, number of flows in intermediate nodes, airtime usage, and physical data rate of each end device [14], [15]. Parameters like buffer capacity, airtime usage are directly affected by the chosen $cwnd$ value, the number of flows in the intermediate node and the physical data rate are independent of the decision made by the CC algorithm. Hence they act as context for which the RL algorithm must be trained.

The unpredictability and dynamic of wireless networks further make it challenging. The existing RL-based algorithms are trained based on partial network information and do not consider the changing dynamics of wireless networks as context. Having a detailed insight into the network would also reduce the number of state and context spaces used to train the algorithm. Therefore the RL-based CC algorithm is modelled as Contextual Markov Decision Process (C, S, A, $M(c)$) where C is called the context space (independent of the actions), S and A are the state and action space respectively. M is the mapping function of context $c \in C$ to an MDP $M(c) = (S, A, p_c(s'|s,a), r_c(s'), \pi_{c0})$, where $p_c(s'|s,a)$ is the transition probability from state s to s' $(s', s \in S, a \in A)$, $r_c(s')$ is the reward obtained when moving from state s to s', and π_{c0} is the initial state distribution for context c. Since the end device is aware of the real-time network information through INT, we consider that the context space is observable.

B. Proposed solutions

Using the real-time network information available through INT significantly reduces the state space, as the algorithm now has a detailed insight into the ongoing changes. In the previous implementations of CC algorithms, it has been proven that using INT information gives better view of the network resulting in simpler and better performing congestion control algorithms [14], [15]. Keeping in mind the challenges addressed in Section III, three multi-context-aware RL-based CC algorithm solutions are proposed as follows:

1) **Fixed contexts**: This is inspired by existing recommendation systems where individual contexts are considered, and since the context space in wireless networks is observable through INT, the RL-model is trained separately for each of these contexts as shown in Figure 1. Though the algorithm addresses the issue of unpredictability of wireless networks by training the models for different contexts, it comes with the disadvantage of large context space. Additionally, it is a tedious task to emulate all the possible contexts, especially for a wireless network, to train the model. If we take the instance of number of flows in the network, different models must be trained for different scenarios of number of flows in the network.
2) **Fixed number of relative contexts**: Instead of considering the exact value of the context, the algorithm is trained for relative context values. The algorithm starts with the stable context level; whenever there is a change in the context, for instance, an increase or decrease in the number of flows or the physical data rate, then it moves to a different context level. The solution drastically reduces the context space and can be easily adapted to any wireless network as it considers only the relative values of the context. The convergence of the algorithm might be longer compared to the first case as each stable context level is different in each scenario.

Figure 1 summarizes the proposed solutions where each color indicates particular context for example, number of traffic flows and each color gradient layer of this color indicates different physical data rates supported by the channel such 6 Mbps, 12 Mbps, etc. These separate

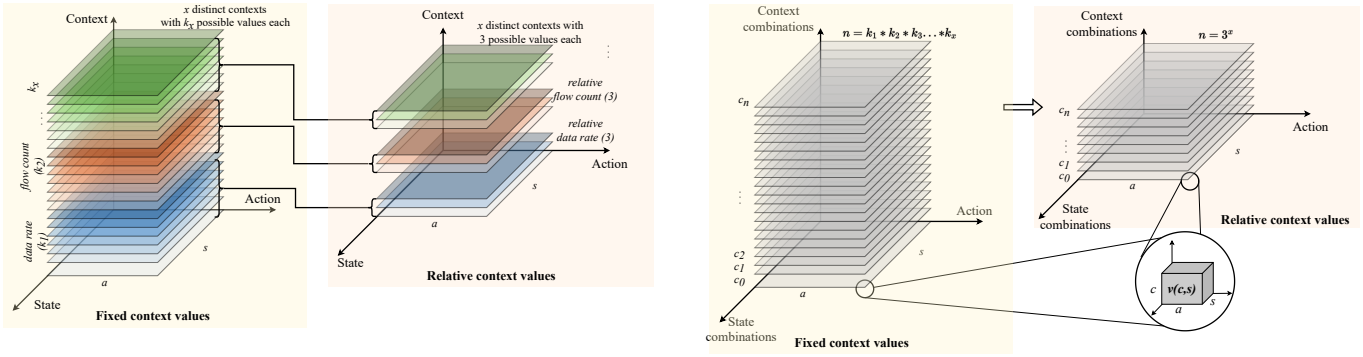

Fig. 1: The figure on the left indicates the compression of contexts to relative contexts which in turn reduces the total number of context values while training (on the right). $v(c, s)$ is the maximum expected future reward (Q value) obtained on taking action a at context c and state s.

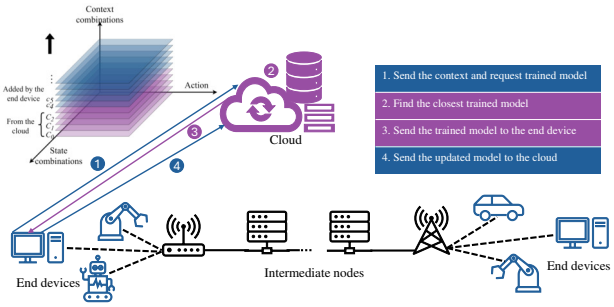

Fig. 2: Cloud-based solution to train multi-context-aware RL-based CC algorithm

Solution	Number of context combinations	Training complexity
Fixed contexts	$n = k_1 * k_2 * ... * k_x$	$\mathcal{O}(n)$
Relative contexts	3^x	$\mathcal{O}(3^x)$
Dynamic contexts	$n = k_1 * k_2 * ... * k_x$	$\mathcal{O}(\frac{n}{m})$

TABLE I: The number of context combinations and training complexity for the proposed multi-context-aware RL CC solutions. x indicates the number of distinct contexts, n indicates the total number of context combinations for x contexts, with k_x possible values each. m indicates the number of trained context combinations received from the cloud.

layers are compressed in relative contexts. The resulting number of context values for fixed contexts solution is more than that of relative contexts solution.

3) **Dynamic number of contexts**: A cloud-based solution is proposed where the end device can dynamically add the context layers and partially retrain on top of the existing trained model obtained from the cloud. The cloud maintains a set of trained models. Whenever the end device requests a trained model for a certain context, the cloud defines to what extent the trained model is different from the requested model and finds the model that is closest to the requested context. The end device can then use this model with a specific exploration rate to customize it to the current context. The exploration rate is dependent on the relative difference between the current context and the context of the trained model from the cloud. The end device posts this locally trained back to the cloud. Along with confining the context space and addressing the unpredictability of the wireless networks, the solution also maintains a homogeneous model and reduces the computational and training expense at each end device. The network architecture used for such a system is represented in Figure 2.

The number of context combinations and training complexity for each of these proposed solutions is listed in the Table I. In fixed contexts case, for x distinct contexts, with k_x possible values for each context, the total number of context combinations is $n = k_1 * k_2 * ... * k_x$. Therefore, the training complexity is equal to $\mathcal{O}(n)$. In the case of relative contexts, for x distinct contexts, there are three possible values for each context. Hence the number of context combinations is 3^x and the training complexity is $\mathcal{O}(3^x)$. The training complexity of the relative context solution is less than the previous solution. For the solution of dynamic contexts, for x distinct contexts, with k_x possible values for each context, assuming the end device receives a trained model from the cloud for m context combinations, the training complexity is now reduced to $\mathcal{O}(\frac{n}{m})$.

V. Q-LEARNING APPROACH FOR FIXED CONTEXT SPACE

As a preliminary test, Epsilon-Greedy Q-learning is used to model multi-context-aware CC algorithm. The Q value equation for CMDP can be formulated as: $Q : (C, S)\text{x}A \rightarrow R$

$$Q'_c[s_t, a_t] \leftarrow Q_c[s_t, a_t] + \alpha(r_c(s_{t+1}))+ \gamma(max_a(Q_c[s_{t+1}]) - Q_c[s_t, a_t])$$

The equation can be further elaborated as,

$$Q'[c_t, s_t, a_t] \leftarrow Q[c_t, s_t, a_t] + \alpha(r_c(s_{t+1}))+ \gamma(max_a(Q[c_{t+1}, s_{t+1}]) - Q[c_t, s_t, a_t]) \quad (1)$$

where α and γ are the learning rate and discount factor respectively [17]. $Q'_c[s_t, a_t]$ and $Q_c[s_t, a_t]$ are the new and

current Q values respectively. $r_c(s_{t+1})$ is the reward for moving from the state s_t to s_{t+1} and $max_a(Q_c[s_{t+1}])$ is the maximum expected future reward obtained from context c_{t+1} and state s_{t+1}. c_{t+1} and s_{t+1} are the context and state in the next time slot ($next\_ts\_context$, $next\_ts\_state$). s_{t+1} is the result of taking action a_t. c_{t+1} is independent of a_t and can be same or different from the previous context. In case of change in context, $max_a(Q_c[s_{t+1}])$ is chosen from the Q-table of the changed context and uses this table until there is change in the context.

To evaluate the feasibility of the proposed solutions, CC is designed for a Wi-Fi network with fixed context (as per system 1 in Section IV-B). The context, state and action spaces are defined below:

- **Context**: for a fixed number of flows in the network at all the times, the physical data rates are fixed to 6, 12, 24 Mbps.
- **State**: change in the packet arrival rate at the intermediate nodes (here access point) measured in four levels, airtime usage of the end device over a fixed timeslot, measured in four levels. Packet arrival rate is the number of packets arriving at the interface of an access point at each instant of time. The increase or decrease in this value is grouped into four levels.
- **Action**: Four action levels are defined, as follows:
  - 0→$cwnd = cwnd + 1$
  - 1→reduce the $cwnd$ by the number of packets in flight
  - 2→increase the $cwnd$ by the number of packets ACKed (num_ACKed)
  - 3→$cwnd = cwnd$
- **Reward**:

$$100 * \frac{num\_ACKed - num\_loss}{cwnd}$$

  where num_loss is the number of packets lost for $cwnd$ packets sent.
- **Q table dimension**: 3x16x4
- $\alpha \leftarrow$0.8 and $\gamma \leftarrow$0.9
- $\epsilon \leftarrow 1$ for training and 0 for testing

The algorithm represented in Algorithm 1, was implemented in Python3.8 programming language using Congestion Control Plane (CCP) framework of MIT [16] on commercially available wireless devices. The algorithm was trained online for different fixed contexts for more than 20000 steps for each context in WiLab-2 testbed.

The trained algorithm was tested on two devices with different physical data rates, 24 Mbps and 6 Mbps, to send a 10 MB payload. As shown in Figure 3, with the CUBIC algorithm, both end devices take the same amount of time to transmit the data, irrespective of their physical data rates. However, with the C-RL algorithm, the end device with the higher physical data rate completed the transfer faster. This difference is also evident in throughput (mentioned in the legend of Figure 3), where the end device 1 achieved better performance with the C-RL algorithm. This proves that the existing CC algorithms

Algorithm 1 Q-learning based context-aware RL (C-RL) CC algorithm with fixed contexts

1: $init\_cwnd \leftarrow 10$
2: TCP (bytes ACKed, packet loss, etc.) and INT report (packet arrival rate, no. of flows, etc.)
3: Set $next\_ts\_context$ and $next\_ts\_state$ from the report
4: Update Q value using equation 1
5: Choose $action$ based on ϵ-greedy method
6: Set $cwnd$ based on the chosen $action$ for next data transfer
7: Go to step 2 to receive TCP and INT report

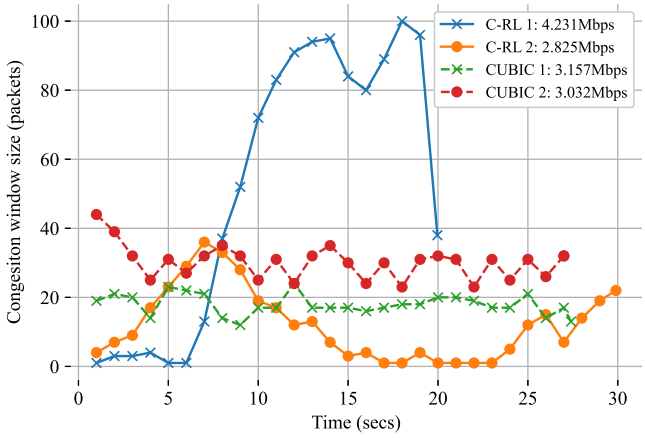

Fig. 3: Behaviour of congestion window size for context-aware RL (C-RL) and CUBIC CC algorithms for end devices 1 and 2 with physical data rates 24 Mbps and 6 Mbps respectively. (Throughput is in the legend.)

are insensitive to contexts like channel quality and competing nodes in the network, which have significant influence on the performance of end device applications. By considering these contexts while designing the RL-based CC algorithms, we can significantly improve the performance of the applications by adapting to wireless network conditions.

VI. Conclusion and Future Work

Private professional wireless networks are increasingly used in industries like manufacturing and energy to meet strict demands for throughput and latency. Future wireless protocols aim to integrate application-network layers, continuous monitoring, and AI adaptation. This paper proposes a multi-context-aware reinforcement learning-based congestion control algorithm, modeled as a contextual MDP, with three RL-based solutions. One solution, using Q-learning, was tested on commercial wireless devices, showing the feasibility of integrating real-time context and AI into existing networks. The proposed solutions will be further tested in the future.

Acknowledgment

This research was partially funded by the Flemish FWO SBO S003921N VERI-END.com project and the Flemish

Government under the “Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen” program.

REFERENCES

[1] Paolucci, F., Cugini, F., Castoldi, P. and Osiński, T., 2021. Enhancing 5G SDN/NFV edge with P4 data plane programmability. IEEE Network, 35(3), pp.154-160.

[2] Haxhibeqiri, J., Isolani, P.H., Marquez-Barja, J.M., Moerman, I. and Hoebeke, J., 2020. In-band network monitoring technique to support SDN-based wireless networks. IEEE Transactions on Network and Service Management, 18(1), pp.627-641.

[3] Haxhibeqiri, J., Seferagic, A., Bhat, R.V., Moerman, I. and Hoebeke, J., 2021, August. Tighter application-network interfacing to drive innovation in networked systems. In Proceedings of the ACM SIGCOMM 2021 Workshop on Network-Application Integration (pp. 53-57).

[4] Miano, S., Bertrone, M., Risso, F., Tumolo, M., & Bernal, M. V. (2018, June). Creating complex network services with ebpf: Experience and lessons learned. In 2018 IEEE 19th International Conference on High Performance Switching and Routing (HPSR) (pp. 1-8). IEEE.

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

[6] Jin, R., Li, J., Tuo, X., Wang, W. and Li, X., 2018. A congestion control method of SDN data center based on reinforcement learning. International Journal of Communication Systems, 31(17), p.e3802.

[7] Xiao, K., Mao, S. and Tugnait, J.K., 2019. TCP-Drinc: Smart congestion control based on deep reinforcement learning. IEEE Access, 7, pp.11892-11904.

[8] Hwang, K., Hsiao, M., Wu, C. and Tan, S., 2005, May. Multi-agent congestion control for high-speed networks using reinforcement co-learning. In International Symposium on Neural Networks (pp. 379-384). Berlin, Heidelberg: Springer Berlin Heidelberg.

[9] Li, Z., Liu, P., Xu, C., Duan, H. and Wang, W., 2017. Reinforcement learning-based variable speed limit control strategy to reduce traffic congestion at freeway recurrent bottlenecks. IEEE transactions on intelligent transportation systems, 18(11), pp.3204-3217.

[10] Sun, J. and Zukerman, M., 2007. An adaptive neuron AQM for a stable internet. In NETWORKING 2007. Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet: 6th International IFIP-TC6 Networking Conference, Atlanta, GA, USA, May 14-18, 2007. Proceedings 6 (pp. 844-854). Springer Berlin Heidelberg.

[11] Midhula, K.S., 2023. An Adaptive Congestion Control Protocol for Wireless Networks Using Deep Reinforcement Learning. IEEE Transactions on Network and Service Management.

[12] Jiang, H., Luo, Y., Zhang, Q., Yin, M. and Wu, C., 2017. TCP-Gvegas with prediction and adaptation in multi-hop ad hoc networks. Wireless Networks, 23, pp.1535-1548.

[13] Yau, K.L.A., Komisarczuk, P. and Teal, P.D., 2012. Reinforcement learning for context awareness and intelligence in wireless networks: Review, new features and open issues. Journal of Network and Computer Applications, 35(1), pp.253-267.

[14] Bhat, R.V., Haxhibeqiri, J., Moerman, I. and Hoebeke, J., 2021, October. Adaptive transport layer protocols using in-band network telemetry and eBPF. In 2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob) (pp. 241-246). IEEE.

[15] Bhat, R.V., Haxhibeqiri, J., Moerman, I. and Hoebeke, J., 2024. Feedback-Based Control Loop Congestion Control Algorithm for Wireless Networks. IEEE Access.

[16] Narayan, A., Cangialosi, F., Raghavan, D., Goyal, P., Narayana, S., Mittal, R., Alizadeh, M. and Balakrishnan, H., 2018, August. Restructuring endpoint congestion control. In Proceedings of the 2018 Conference of the ACM Special Interest Group on Data Communication (pp. 30-43).

[17] Watkins, C.J. and Dayan, P., 1992. Q-learning. Machine learning, 8, pp.279-292.