DRL-aided Coded Computation Offloading Framework for Mode-2 of 5G NR-V2X

Pritom Das, *Rajib Paul, Young-June Choi**

Department of Artificial Intelligence, Ajou University
pritom@ajou.ac.kr, *rajib@ajou.ac.kr, **choiyj@ajou.ac.kr

Abstract

In Mode-2 of 5G NR-V2X, computation offloading decisions have to made by the user vehicles autonomously to satisfy latency and reliability requirements. In this paper, we propose a coded edge computing (CEC) based computation offloading framework that minimizes the end-to-end latency and the cost of edge computing. Powered by deep reinforcement learning, the proposed approach is able to cope with the dynamic environment characterized by high mobility, time-varying sidelink resources and heterogeneous edge resources. We also present the simulation results and comparison of the proposed method to baseline approaches.

Ⅰ. Introduction

5G new radio Vehicle-to-Everything (5G NR-V2X) services involves computations under rigorous latency and reliability requirements. Multi-access edge computing (MEC) provides improvement in latency by offloading computations to the servers at the network edge, However, heterogeneous computing resources on the servers cause straggler effect. To remedy this, many researchers have considered coded edge computing (CEC) where computation are split and encoded into subtasks before they are sent to several servers [1]. Although promising, CEC has not been considered in the context of 5G NR-V2X, particularly mode-1, where no gNB (base station defined in 5G) is involved in managing radio resources, makes offloading decisions or acts as a mediator among the vehicles.

In this paper, we consider that the user vehicle (UV) must make the offloading decisions autonomously. Specifically, the UV must maintain the recovery threshold i.e. the minimum number of subtasks to ensure reliability, the number of coded subtasks and select the servers while considering the available server vehicles (SV) within its coverage area, computing resources on the servers as well as the sidelink resources, all of which are time-varying. The decision-making is facilitated by a deep Q learning (DQN) based algorithm, capable of adaptively learning to robustly address the dynamic and mobile environment. We formulate the problem as an optimization of end-to-end computation latency and the cost of edge computing.

Ⅱ. System Model and Problem Formulation

We consider that the UV and N number of SVs moving along a highway and their speeds are defined by a truncated gaussian distribution. The distance of the SVs, d_n and the available computing resources on them, f_n is known to the UV through cooperative messaging. In each time-frame, UV needs to perform a multiplication of two matrices, **A** and **B**. UV employ PolyDot coding to split and encode the computation [2].

Firstly, **A** is split so that it contains t submatrices as rows and s submatrices as columns, such as $st = \varphi$, where φ is the number of uncoded subtasks, whereas B contains s submatrices as rows and t submatrices as columns. We obtain the recovery threshold as $\psi_{thres} = t^2(2s-1)$. Considering $t = 1$, $\psi_{thres} = (2s-1)$, given that $s = \varphi$. Since $\psi_{thres} \leq N$, therefore $1 \leq \varphi \leq \frac{N+1}{2}$. These subtasks are then encoded to generate ψ number of coded subtasks to be offloaded to ψ SVs. Since ψ must be smaller than the number of available subchannels, K_{av}, therefore $\psi_{thres} \leq \psi \leq \min(N, K_{av})$.

Computations are performed in two modes: local and edge computing. T_{total} is the time required for UV to perform the computation locally. In the edge computing mode, for the n-th SV, the end-to end latency, T_n^{E2E} is the summation of the transmit time, T_n^{trans}, time to compute the task on the SV, T_n^{comp}, and the time to download the outputs from the SV to UV, T_n^{down}. Since the output size is much smaller compared to the coded subtasks. T_n^{down} is negligible. Therefore, we obtain, $T_n^{E2E} = T_n^{trans} + T_n^{comp}$. The total latency to receive the outputs from all the servers is the time required for the slowest of the SVs. Thus, the end-to-end edge computing latency for each time-step is $T_{CEC} = \max(\lambda_1 T_1^{E2E}, .. \lambda_N T_N^{E2E}), \lambda_n \epsilon \Lambda$, where $\Lambda = \{\lambda_1 .. \lambda_N\}$ denotes the SVs selected by the UV. If the n-th SV is selected then $\lambda_n = 1$, otherwise $\lambda_n = 1$. Computations are offloaded to the SVs only when $T_{CEC} < T_{local}$. Now, we can obtain the overall end-to-end latency of the system by

$$T_{system} = \begin{cases} T_{CEC}, & if\ T_{CEC} < T_{local} \\ T_{local}. & otherwise \end{cases}$$

SVs offer computing services for a price. If the cost at computing on the n-th SV is E_n, then the total cost of edge computing is $E_{total} = \sum_{n=1}^{N} E_n$.

Our objective is to minimize the overall end-to-end latency of the system and the computing cost. Thus we obtain the utility of the system, $U_{CEC} = -\Omega_1 log_{10} T_{system} - \Omega_2 log_{10} E_{total}$, where Ω_1 and Ω_2 denote the priority of the utility components. Now we can define our objective function as the maximization of the total utility of the UV by selecting φ and Λ, given that $d_n \leq R_{cov}$, where R_{cov} is the coverage radius of UV:

$$\max_{\varphi,\ \Lambda} U_{CEC}, \quad (1)$$

subjected to

$$1 \le \varphi \le \frac{N+1}{2}, \quad (2)$$

$$\psi_{thres} \le \psi \le \min(N, K_{av}), \quad (3)$$

$$d_n \le R_{cov}, \quad (4)$$

Ⅲ. DQN-based Offloading Algorithm

We model the optimization problem in Eqs (1)-(4) as a Markov decision process (MDP) problem with the UV acting as an agent and the following state and action spaces and reward function:

State space: The state space includes the information required for the UV to make offloading decision. Thus the state space $S(t) = \{D, F, K_{av}$, where $D = \{d_n\}$ and $F = \{f_n\}$.

Action space: Action space is defined as $A(t) = \{\varphi,\ \Lambda\}$.

Reward function: We define the instant reward as $r(t) = U_{CEC} + \sum_{n=1}^{N} \lambda_n C_n$, where $C_n = c$, if the computation is done within the time the n-th SV stay within R_{cov}. Otherwise, $C_n = 0$. Over time, the agent learns so that the cumulative reward $R_C + \sum_{t=1}^{T} r(t)$, where T is the number of total time steps.

The agent follows an ε-greedy policy. At each timestep, the agent acquires the sate space information, takes an action a_t, and obtains the reward $a_t$and the next state S'. This information i.e., $\{S_t, a_t, S_t, r_t\}$ is then stored in the experience replay. The agent updates the Q-parameter by randomly extracting a batch of experiences from the replay and calculating the mean squared difference between the Q-value $Q(S, a, \theta)$ and target Q-value, $r + \gamma_{dqn} \max_a Q(S', a', \theta)$, where γ_{dqn} is the discount factor. This process is iterated for a number of episodes, each of which consists of a number of steps.

IV. Performance Analysis

Fig. 1 shows the convergence performance of the proposed approach. It is evident that the convergence is fastest when the mean speed is lowest and slowest when the mean speed is the highest. The reason is that as the mean speed increases, the variance in the speeds of the UV and SVs also increases. That increases the unpredictability of the environment leading to slower convergence.

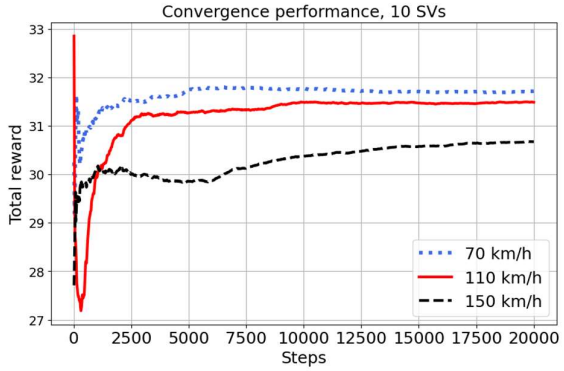

Figure 1: Training reward for 10 SVs, mean speed = 70, 110, 150 km/h

Fig. 2 and Fig. 3 compare the proposed method to other commonly applied methods such as local only computation (LO), edge only computation (EO) and random coding (RC) i.e., φ and Λ are randomly selected. Results shows that better latency and cost performance are achieved with the proposed method.

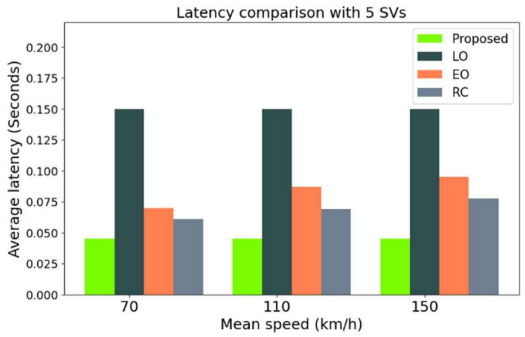

Figure 2: Comparison of average latency, 5 SVs, mean speed = 110 km/h

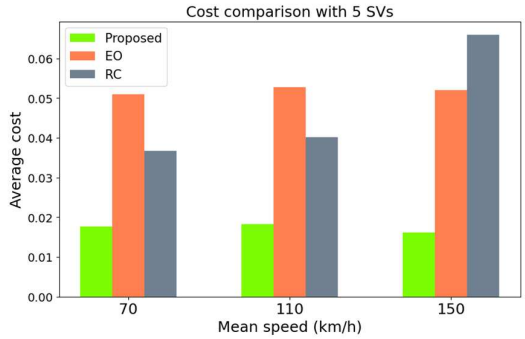

Figure 3: Comparison of average cost, 5 SVs, mean speed = 110 km/h

V. Conclusion

In this paper, we presented a deep reinforcement learning based coded computation offloading approach for mode-2 of 5G NR-V2X. Results show that the proposed approach can adapt with the dynamic vehicular environment and outperform widely used methods. In future, we aim to extend this work to multiuser scenario with optimized energy consumption model.

REFERENCES

[1] A. Asheralieva and D. Niyato, "Fast and Secure Computational Offloading With Lagrange Coded Mobile Edge Computing," in IEEE Transactions on Vehicular Technology, vol. 70, no. 5, pp. 4924-4942, May 2021, doi: 10.1109/TVT.2021.3070723.

[2] S. Dutta, M. Fahim, F. Haddadpour, H. Jeong, V. Cadambe and P. Grover, "On the Optimal Recovery Threshold of Coded Matrix Multiplication," in IEEE Transactions on Information Theory, vol. 66, no. 1, pp. 278-301, Jan. 2020, doi: 10.1109/TIT.2019.2929328.