Cooperative Task Offloading with Multi-agent Deep Reinforcement Learning

Xinyao Qiu
Yuqi Mai

July 1, 2023

Problem

- To meet the growing computing needs of users, mobile edge computing (MEC) is a promising technology that can improve the computing experience of electronic devices by offloading computation-based tasks to MEC servers located near the base station. MEC task offloading has become a viable solution to meet various EDs computing needs and improve the quality of end-user experience.
- In this project, we simplify the MEC problem as video task (VT) processing. When we compress a video, two options can be chosen, which are processed locally or offload video tasks to the MEC server for cloud computing.

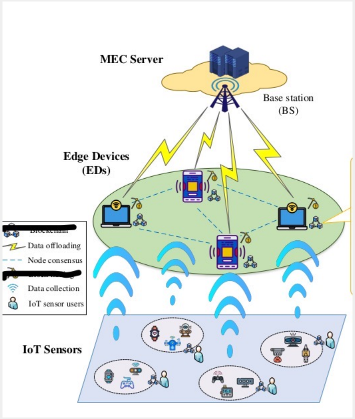

Figure: cooperative task offloading system

Motivation

- Our objective here is to find the optimal utility for the whole system consisting of numbers of EDs and MEC, and we can control different parameters (action) to affect the system (state). So we apply reinforcement learning to handle the problem.
- Specifically, we use the Deep deterministic policy gradient algorithm (DDPG) for the problem.
- We compare the reward function for different environment parameters. And we also compare the performance of DDPG in this problem with a few other algorithms, such as advantage actor critic (A2C) and proximal policy optimization (PPO) by comparing their final rewards.

.

Method

- The deep deterministic policy gradient (DDPG) (Lillicrap et al. 2016) has a very similar structure to Actor-Critic (AC) model, which contains two deep neural networks, an actor-network, and a critic network. The actor-network takes in the current state and outputs selected actions while the critic network takes in the state and the actions and evaluate the Q-value.
- Critic: Q-network $Q_w(s, a)$: trained with mini-batch and temporal-difference.
- Deterministic policy-network $\mu_\theta(x)$: trained with loss function and output continuous actions.

$$L(\theta) = -\sum_{\mathrm{s}_b \in minibatch} Q_w(s_b, \mu_\theta(s_b)) \tag{1}$$

- We use a modified DDPG algorithm in our project, MADDPG, which stands for Multi-Agents DDPG. The major difference between MADDPG and DDPG is in MADDPG, the critic neural network takes in multiple state information sensed by different agents to evaluate Q-value.

Model

- Task offloading policy $\boldsymbol{X} = \{x_n^k | n \in \boldsymbol{N},\ k \in \boldsymbol{K}\}$

$$x_n^k = \begin{cases} 1 & \text{, MEC mode \& using channel } k \\ 0 & \text{, otherwise.} \end{cases}$$

  Further, we denote $x_n = \sum_{k \in \boldsymbol{K}} x_n^k$.

- Transmit power policy $\boldsymbol{P} = \{p_n^k | 0 \le p_n^k \le p_{max},\ n \in \boldsymbol{N},\ k \in \boldsymbol{K}\}$.
- Local computational resource allocated to VT_n $\boldsymbol{F} = \{f_n^k | f_{min} \le f_n^k \le f_{max},\ n \in \boldsymbol{N},\ k \in \boldsymbol{K}\}$.
- Utility $R_n = \lambda_1 \frac{\phi_n^{loc} - \phi_n}{\phi_n^{loc}} + \lambda_2 \frac{E_n^{loc} - E_n}{E_n^{loc}}$. The problem can be formulated as

$$\begin{aligned} \underset{\boldsymbol{X}, \boldsymbol{P}, \boldsymbol{F}}{\text{maximize}} \quad & \frac{1}{N} \sum_{n \in \boldsymbol{N}} R_n && (2a) \\ \text{subject to} \quad & x_n^k \in \{0, 1\}, \quad x_n \le 1, && (2b) \\ & 0 \le p_n^k \le p_{max}, && (2c) \\ & T_n \le \tau_n, \quad \phi_n \ge \epsilon_n, && (2d) \\ & f_{min} \le f_n^k \le f_{max}. && (2e) \end{aligned}$$

State and Action

- $\mathcal{S}(t) = \{S_{task}(t), S_{channel}(t), S_{pow}(t), S_{res}(t)\}$.
  - **Task state:** $S_{task}(t) = [S_n(t), C_n(t)]$, $n \in \boldsymbol{N}$. $S_n(t)$ represents the task size of VT_n, and $C_n(t)$ represents the required CPU cycle to compute VT_n.
  - **Mode & channel selection state:** $S_{channel}(t) = \boldsymbol{X}(t)$
  - **Power transmission state:** $S_{pow}(t) = \boldsymbol{P}(t)$
  - **Local resource allocation state:** $S_{res}(t) = \boldsymbol{F}(t)$,
- $\mathcal{A}(t) = \{x_n^k(t), k(t), f_n^k(t), \beta_n^k(t)\}$.
  - **Task offloading decision:** $x_n^k(t) \in \{0, 1\}$**.**
  - **Channel selection:** $k(t) = [1, 2, ..., K]$**.**
  - **Local computational resource allocation:** $f_n^k(t) \in [f_{min}, f_{max}]$**.**
  - **Compression rate selection:** $\beta_n^k(t) \in [0, 1]$**.**

Experiment and Result

We use the dataset in [1], which is the mobile wireless dataset provided by Shanghai Telecom.

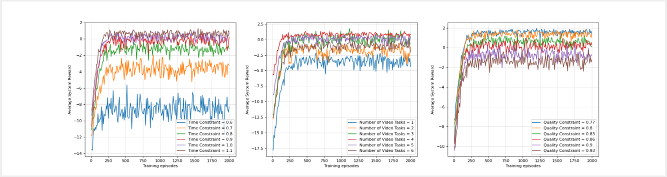

Figure: Comparison of average system rewards with different parameters.

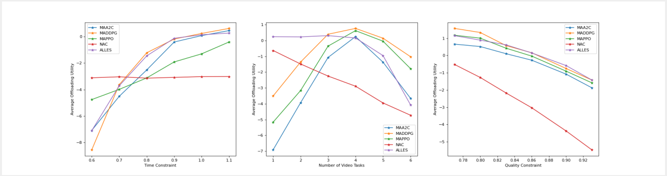

Figure: Comparison of average system rewards with different algorithms.

Conclusion

- **Parameters:** For time constraint, increase; for number of video tasks, increase and decrease; for quality constraint, decrease.
- **Algorithms:** MADDPG can get largest reward.
- **Different agents:** Change the value of power s_n^k of one agent, the energy and quality of it would increase, but other agents' energy would increase and reward would decrease.

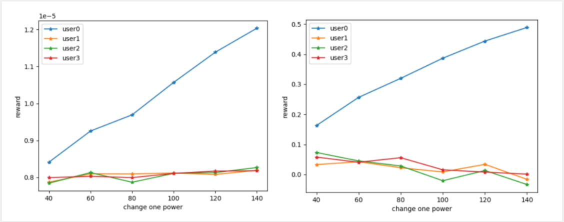

Figure: Change one agent's power

Appendix

Modeling Details

Define $\beta_n \in [0,1]$ as the compression rate of VT_n, W as the bandwidth of channels, σ^2 as the background noise variance, α as computation capacity, κ as the enegy coefficient, ν as the quality coefficient, θ as the normalization coefficient. The formulations are

$$T_n^{loc} = \frac{\beta_n c_n}{f_n}, \quad E_n^{loc} = \kappa S_n \beta_n (f_n)^2, \quad \phi_n^{loc} = \nu_{loc} log_2 \left(1 + \frac{\beta_n}{\theta_{loc}}\right)$$

$$T_n^{off} = \frac{\beta_n s_n}{r_n} + \frac{\beta_n c_n}{\alpha}, \quad E_n^{off} = \sum_{k \in \boldsymbol{K}} p_n^k T_n^{off}, \quad \phi_n^{off} = \nu_{off} log_2 \left(1 + \frac{\beta_n}{\theta_{off}}\right)$$

MEC mode's data transmission rate r_n is calculated as

$$r_n = W log_2 \left(1 + \frac{p_n^k h_n^k}{\sigma^2 + \sum_{i \in \boldsymbol{N}, i \neq n} x_i^k p_i^k h_i^k}\right)$$

$$T_n = (1 - x_n) T_n^{loc} + x_n T_n^{off}$$

$$E_n = (1 - x_n) E_n^{loc} + x_n E_n^{off}, \quad \phi_n = (1 - x_n) \phi_n^{loc} + x_n \phi_n^{off}$$

References

- D. Nguyen, M. Ding, P. Pathirana, A. Seneviratne, J. Li, and V. Poor, "Cooperative task offloading and block mining in blockchain-based edge computing with multi-agent deep reinforcement learning," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2021.

- J.R, Y.Guo, D.Zhang, D.Liu, Q.Zhang, and Y.Xue, "Distributed and efficient object detection in edge computing: Challenges and solutions," *IEEE Network*, vol. 32, no. 6, pp. 137–143, 2018.

- L.Xiao, Y.Ding, D.Jiang, J.Huang, D.Wang, J.Li, and P. H. Vincent, "A reinforcement learning and blockchain-based trust mechanism for edge networks," *IEEE Trans. Commun.*, vol. 68, no. 9, pp. 5460–5470, Sept. 2020.

- B. Yang, X. Cao, J. Bassey, X. Li, and L. Qian, "Computation offloading in multi-access edge computing: A multi-task learning approach," *IEEE Trans. Mob Comupt.*, pp. 1–1, 2021. doi:10.1109/TMC.2020.2990630.

THANKS!