QOCO: A QoE-Oriented Computation Offloading Algorithm based on Deep Reinforcement Learning for Mobile Edge Computing

Iman Rahmati, Hamed Shah-Mansouri, and Ali Movaghar

Sharif University of Technology, Tehran, Iran

December 3, 2023

Overview

Table of Contents

Edge Computing as a Solution

Edge Computing

- Computation at the edge of the network.

Computation Offloading

- Enhances the capacity of mobile devices.

Benefits

- Reduced task performance delay.
- Extended mobile device (MD) battery life.
- Enhanced Quality of Experience (QoE).

Challenges

- Dynamic network conditions.
- Heterogeneous device capabilities.

Table of Contents

Contribution

Research Focus: Computation Task Offloading Problem in MEC

- Task Processing Deadlines
- Energy Constraints

Key Contributions:

- Formulated the computation task offloading problem as a finite and discret Markov Decision Process (MDP) to maximize the expected long term QoE for each MDs.
- Proposed a QoE-oriented computation offloading (QOCO) algorithm based on Deep Reinforcement Learning (DRL) to empower each MD to make offloading decisions independently.
- Conducting comprehensive experiments to evaluate the performance of QOCO compared with several benchmark methods.

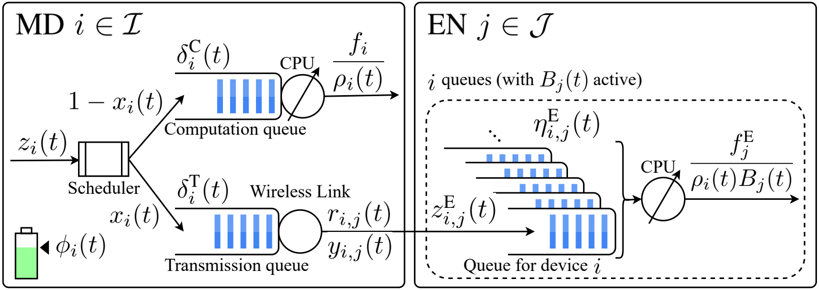
Table of Contents

System Model

We consider

- the set of MD $\mathcal{I} = \{1, 2, ..., I\}$
- the set of EN $\mathcal{J} = \{1, 2, ..., J\}$
- set of time slots $\mathcal{T} = \{1, 2, \ldots, T\}$
- each MD $i \in \mathcal{I}$ are connected to each EN $j \in \mathcal{J}$ with it's wireless interface.
- each time slot consider as a diuration of time.

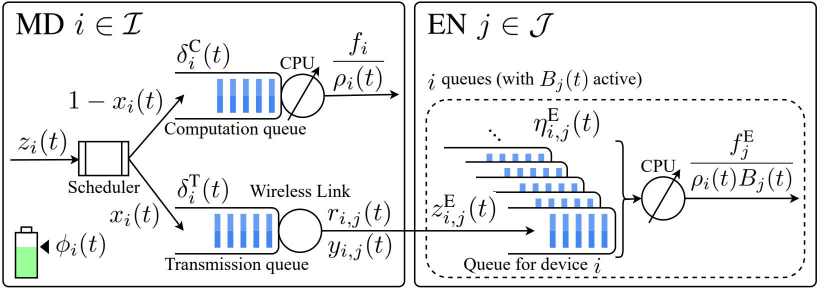
System Model



Task

- $z_i(t)$: arrived task in MD
- $\lambda_i(t)$: Required CPU cycles of task
- $\rho_i(t)$: Required CPU cycles of task
- $\Delta_i(t)$: Required CPU cycles of task

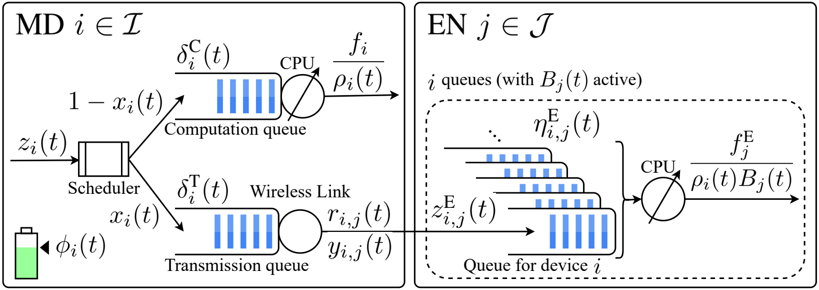
System Model



Offloading

- $x_i(t)$: Offloading Decision
- $y_{i,j}(t)$: Offloading Target

System Model



Queues' Information

- $\delta_i^C(t)$ Comption Queue in MD
- $\delta_i^T(t)$ Transmition Queue in MD
- $\eta_{i,j}^E(t)$ MD Comption Queue in EN

Communication Model

- $\delta_i^T(t)$: MD i Transmission Queue Waiting Time

$$\delta_i^T(t) = \left[\max_{t' \in \{0,1,\ldots,t-1\}} l_i^T(t') - t + 1 \right]^+ \tag{1}$$

- $l_i^T(t)$: Task Transmited/Dropped Time Slot

$$l_i^T(t) = \min \left\{ t + \delta_i^T(t) + \lceil D_i^T(t) \rceil - 1, t + \Delta_i(t) - 1 \right\} \tag{2}$$

- $D_i^T(t)$: Task Transmission Time

$$D_i^T(t) = \sum_{\mathcal{J}} y_{i,j}(t) \frac{\lambda_i(t)}{r_{i,j}(t)\tau} \tag{3}$$

- $E_i^T(t)$: Task Transmission Energy Consumption

$$E_i^T(t) = D_i^T(t) p_i^T(t) \tau \tag{4}$$

Computation Model

Local Execution:

- $\delta_i^C(t)$: MD i Computation Queue Waiting Time

$$\delta_i^C(t) = \left[\max_{t' \in \{0,1,\ldots,t-1\}} l_i^C(t') - t + 1 \right]^+ \tag{5}$$

- $l_i^C(t)$: Task Executed/Dropped Time Slot

$$l_i^C(t) = \min \left\{ t + \delta_i^C(t) + \lceil D_i^C(t) \rceil - 1, t + \Delta_i(t) - 1 \right\} \tag{6}$$

- $D_i^C(t)$: Task Execution Time

$$D_i^C(t) = \frac{\lambda_i(t)}{f_i \tau / \rho_i(t)} \tag{7}$$

- $E_i^L(t)$: Task Execution Energy Consumption

$$E_i^L(t) = D_i^C(t) p_i^C \tau \tag{8}$$

Computation Model

Edge Execution:

- $\eta^E_{i,j}(t)$: MD i Queue Backlog at EN j

$$\eta^E_{i,j}(t) = \left[\eta^E_{i,j}(t-1) + \lambda^E_{i,j}(t) - \frac{f^E_j}{\rho_i(t)B_j(t)} - \omega_{i,j}(t)\right]^+ \quad (9)$$

- $\hat{l}^E_{i,j}(t)$: Task Execution Start Time Slot at EN j

$$\hat{l}^E_{i,j}(t) = \max\{t, \max_{t' \in \{0,1,\ldots,t-1\}} l^E_{i,j}(t') + 1\} \quad (10)$$

$$\sum_{t'=\hat{l}^E_{i,j}(t)}^{l^E_{i,j}(t)} \frac{f^E_j}{\rho_i(t)B_j(t')} \geq \lambda^E_{i,j}(t) > \sum_{t'=\hat{l}^E_{i,j}(t)}^{l^E_{i,j}(t)-1} \frac{f^E_j}{\rho_i(t)B_j(t')} \quad (11)$$

Computation Model

Edge Execution:

- $D_{i,j}^{E}(t)$: Task Execution Time at EN j

$$D_{i,j}^{E}(t) = \frac{\lambda_{i,j}^{E}(t)\rho_i(t)}{f_j^{E}\tau / B_j(t)} \tag{12}$$

- $E_{i,j}^{E}(t)$: Task Execution Energy Consumption at EN j

$$E_{i,j}^{E}(t) = \frac{D_{i,j}^{E}(t) p_j^{E}\tau}{B_j(t)} \tag{13}$$

- $E_i^{I}(t)$: MD i Standby Energy Consumption

$$E_i^{I}(t) = D_{i,j}^{E}(t) p_i^{I}\tau \tag{14}$$

- $E_i^{O}(t)$: Overall Offloading Energy Consumption

$$E_i^{O}(t) = E_i^{T}(t) + \sum_{\mathcal{J}} E_{i,j}^{E}(t) + E_i^{I}(t). \tag{15}$$

Table of Contents

Markov Decision Process

State Space:

$$\boldsymbol{s}_i(t) = \Big(\lambda_i(t), \delta_i^C(t), \delta_i^T(t), \boldsymbol{\eta}_i^E(t-1)^{\circ}, \phi_i(t)^{\times}, \mathcal{H}(t)^{\diamond}\Big) \qquad (16)$$

$^{\diamond}\mathcal{H}(t)$: Edge Load History Matrix

$^{\times}\phi_i(t)$: MD Battery Level

$$\left[\mathcal{S} = \Lambda \times T^2 \times \mathcal{U} \times 3 \times I^{T^{\mathsf{s}} \times J}\right]$$

Action Space:

$$\boldsymbol{a}_i(t) = (x_i(t), \boldsymbol{y}_i(t)^{*}) \qquad (17)$$

$^{*}\boldsymbol{y}_i(t) = (y_{i,j}(t), j \in \mathcal{J})$

Markov Decision Process

QoE Function:

[Delay]

$$\mathcal{D}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t)) = (1 - x_i(t))\Big(l_i^{\mathsf{C}}(t) - t + 1\Big) + \\ x_i(t)\left(\sum_{\mathcal{J}} \sum_{t'=t}^{T} \mathbb{1}\big(z_{i,j}^{\mathsf{E}}(t') = z_i(t)\big) l_{i,j}^{\mathsf{E}}(t') - t + 1\right) \tag{18}$$

[Energy]

$$\mathcal{E}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t)) = (1 - x_i(t)) E_i^{\mathsf{L}}(t) + \\ x_i(t)\left(\sum_{\mathcal{J}} \sum_{t'=t}^{T} \mathbb{1}\big(z_{i,j}^{\mathsf{E}}(t') = z_i(t)\big) E_i^{\mathsf{O}}(t)\right) \tag{19}$$

Markov Decision Process

[Cost]

$$\mathcal{C}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t)) = \phi_i(t)\, \mathcal{D}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t)) + (1 - \phi_i(t))\, \mathcal{E}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t)) \tag{20}$$

[QoE]

$$\boldsymbol{q}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t)) = \begin{cases} \mathcal{R}^* - \mathcal{C}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t)) & \text{if task } z_i(t) \text{ processed,} \\ -\mathcal{E}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t)) & \text{if task } z_i(t) \text{ dropped,} \end{cases} \tag{21}$$

*$\mathcal{R}$: A Constant Reward for Completed Task

QoE Optimization Problem

$$^{\circ}\pi_i^* = \text{arg}\ \max_{\pi_i} \mathbb{E}\left[\sum_{t\in\mathcal{T}} {}^{*}\gamma^{t-1} \boldsymbol{q}_i(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t) \middle| \pi_i\right] \tag{22}$$

$^{\circ}\pi_i^*$: Optimal Policy

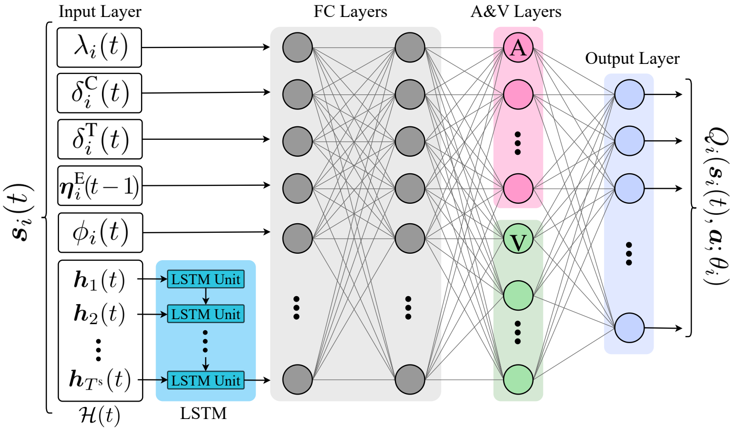
$\Rightarrow$ maximizes the long-term QoE

$^{*}\gamma$: Discount Factor

$\Rightarrow$ balance between instant QoE and long-term QoE

Table of Contents

DQN-based solution



- LSTM: ENs Load Level Prediction
- FC: State-Action Q-Value Mapping
- A&V: Dueling-DQN Approach for Q-Value Estimation

QOCO Algorithm

Structure:

- ENs help MDs to traine their neural networks.
- The algorithms to be executed at MD and EN
- Training neural networks with MD experiences
  - **(i.e., state, action, QoE, next state)**
- Mapping Q-values to each state-action pair

In detail:

- MD's Neural Network in EN:
  - **Replay Memory:** Records the observed experience of MD.
  - **Evaluation Network:** Responsible for action selection. $Q_i^{\mathsf{E}}(\boldsymbol{s}_i(t), \boldsymbol{a}; \theta_i^{\mathsf{E}})$
  - **Target Network:** Characterizes the target Q-values. $Q_i^{\mathsf{T}}(\boldsymbol{s}_i(t), \boldsymbol{a}; \theta_i^{\mathsf{T}})$
- Updating the evaluation network parameter vector:
  - Minimization of the difference between the Q-values under evaluation and target networks.

QOCO Algorithm

Offloading Decision Algorithm at MD:

- MD sends an *UpdateRequest* to EN.
- Receive network parameter vector θ_i^{E} from EN
- Select an Action for each Computation Task based on

$$\boldsymbol{a}_i(t) = \begin{cases} \text{arg } \max_{\boldsymbol{a} \in \mathcal{A}} Q_i^{\mathsf{E}}(\boldsymbol{s}_i(t), \boldsymbol{a}; \theta_i^{\mathsf{E}}), & \text{with p}(1-\boldsymbol{\epsilon}) \\ \text{pick an random action from } \mathcal{A}, & \text{with p}(\boldsymbol{\epsilon}) \end{cases} \tag{23}$$

- Observes a set of QoEs
- Send Experience $(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t), \boldsymbol{q}_i(t), \boldsymbol{s}_i(t+1))$ to EN

QOCO Algorithm

Training Process Algorithm at EN:

- Stores the Receives Experience $(\boldsymbol{s}_i(t), \boldsymbol{a}_i(t), \boldsymbol{q}_i(t), \boldsymbol{s}_i(t+1))$
- Calculates the Q-value given the MD experience

$$\hat{Q}^{\mathsf{T}}_{i,n} = \boldsymbol{q}_i(n) + \gamma Q^{\mathsf{T}}_i(\boldsymbol{s}_i(n+1)), \tilde{\boldsymbol{a}}_n; \theta^{\mathsf{T}}_i) \tag{24}$$

- optimal action for the state

$$\tilde{\boldsymbol{a}}_n = \text{arg} \max_{\boldsymbol{a} \in \mathcal{A}} Q^{\mathsf{E}}_i(\boldsymbol{s}_i(n+1), \boldsymbol{a}; \theta^{\mathsf{E}}_i) \tag{25}$$

- Updating netwrok based on loss function

$$L(\theta^{\mathsf{E}}_i, \hat{\mathbf{Q}}^{\mathsf{T}}_i) = \frac{1}{|\mathcal{N}|} \sum_{n \in \mathcal{N}} \left(Q^{\mathsf{E}}_i(\boldsymbol{s}_i(n), \boldsymbol{a}_i(n); \theta^{\mathsf{E}}_i) - \hat{Q}^{\mathsf{T}}_{i,n} \right)^2 \tag{26}$$

Table of Contents

Simulation Settings

Parameter	**Value**
Computation capacity of MD f_i	2.6 GHz
Computation capacity of EN f_j^{E}	42.8 GHz
Transmission capacity of MD $r_{i,j}(t)$	14 Mbps
Task arrival rate	150 Task/Sec
Size of task $\lambda_i(t)$	$\{1.0, 1.1, \ldots, 7.0\}$ Mbits
Required CPU cycles of task $\rho_i(t)$	{0.197,0.297,0.397}G/Mbits
Deadline of task Δ_i	10 time slots (1 Sec)
Battery level of MD $\phi_i(t)$	{25, 50, 75} Percent
Computation power of MD p_i^{C}	$10^{-27}(f_i)^3$
Computation power of EN p_j^{E}	5 w
Transmission power of MD p_i^{T}	2.3 w
Standby power of MD p_i^{I}	0.1 w

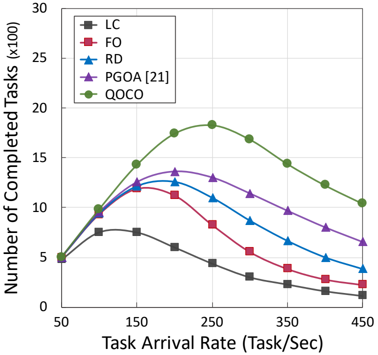
50 MD and 5 EN
1000 Episode and 100 Time Slot

Performance Comparison

Benchmark Methods:

- Local Computing (LC): MDs execute all of their computation tasks using their own computing capacity
- Full Offloading (FO): MDs dispatches all of their computation tasks to ENs and selects their offloading target randomly
- Random Decision (RD): MDs randomly makes offloading decisions and selects the offloading target
- PGOA[21]: A distributed optimization algorithm designed for delay-sensitive tasks in an environment where MDs interact strategically with multiple ENs.

Performance Comparison

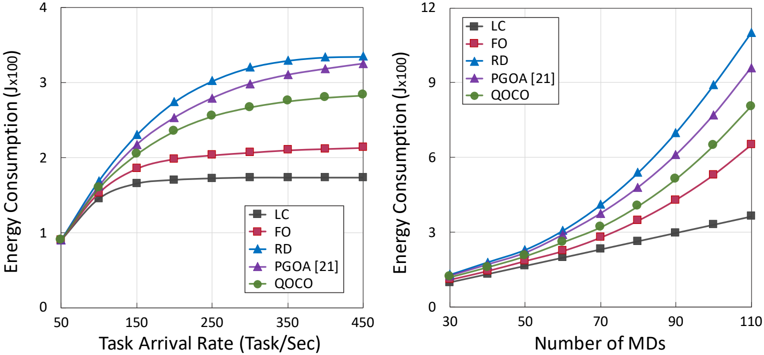


(a) (b)

Number of completed tasks under different computation loads:

- (a) task arrival rate
- (b) the number of MDs

Performance Comparison

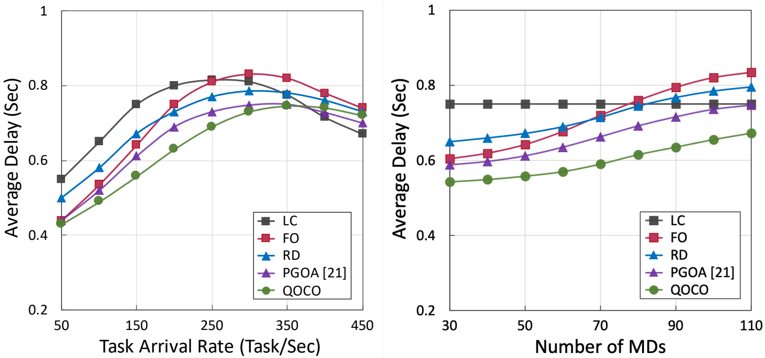


(a) (b)

Overall energy consumption under different computation loads:

- (a) task arrival rate
- (b) the number of MDs

Performance Comparison

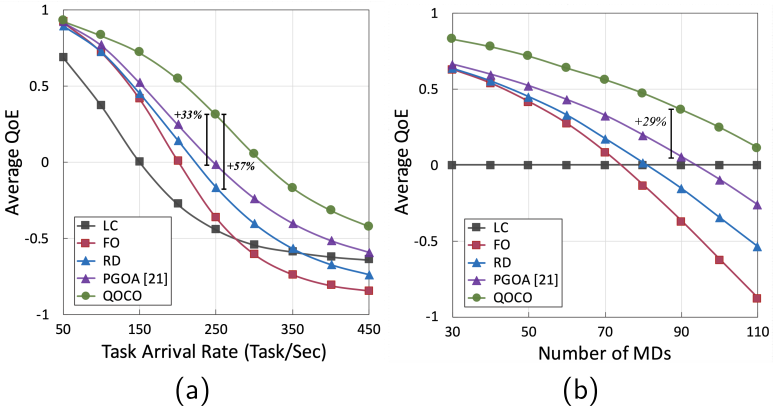


(a) (b)

Average delay under different computation loads:

- (a) task arrival rate
- (b) the number of MDs

Performance Comparison



Average **QOE** under different computation loads:

- (a) task arrival rate
- (b) the number of MDs

Table of Contents

Conclusion

- We formulated an optimization problem that aims to maximize the QoE of each MD individually.
  - The QoE reflects the energy consumption and task completion delay.
- Empowering MDs to make offloading decisions.
  - (without relying on knowledge about task models or other MDs' offloading decisions)
- Adapts to the uncertain dynamics of load levels at ENs.
  - Effectively manages the ever-changing system environment.

Future Work:

- Extending the task model by considering interdependencies among tasks.
  - This can be achieved by incorporating a **task call graph representation** to develop dependency among task partitions.
- Enabling MDs to take advantage of federated learning techniques in the training process.
  - This allows MDs to collectively contribute to improving the offloading model and enable continuous learning when new MDs join the network.