三峡大学
CHINA THREE GORGES UNIVERSITY

IEEE MASS 2020

Multi-Agent Reinforcement Learning for Cooperative Edge Caching in Internet of Vehicles

Author : Kai Jiang[1], Huan Zhou[1], Deze Zeng[2], Jie Wu[3],

Reporter : Kai Jiang

1. China Three Gorges University, China
2. China University of Geosciences, China
3. Temple University, USA

Contents

1 Introduction

2 Network Architecture

3 Problem Formulation

4 Problem Solving

5 Performance Evaluation

6 Conclusion

Contents

Next Part

1 *Introduction*

2 *Network Architecture*

3 *Problem Formulation*

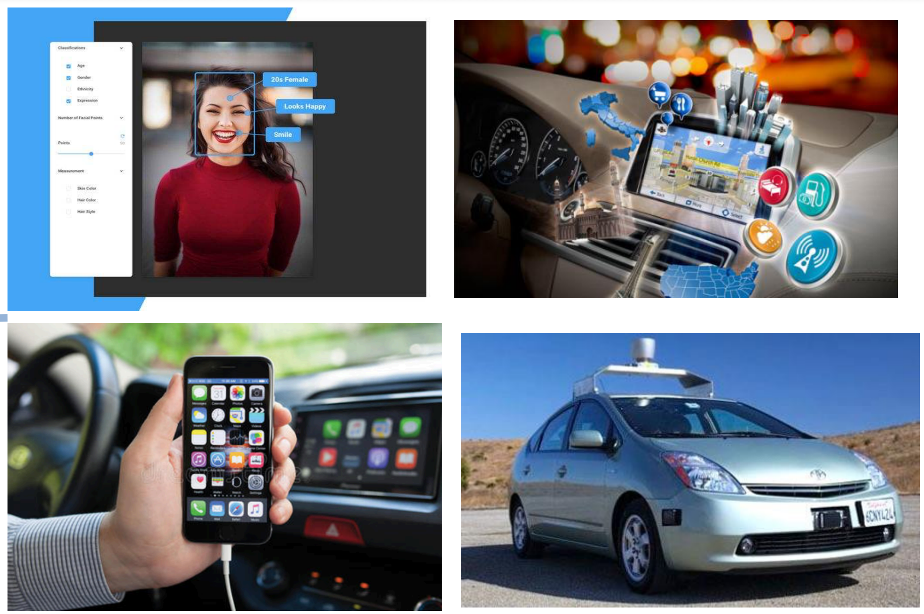
4 *Problem Solving*

5 *Performance Evaluation*

6 *Conclusion*

Flourishing Vehicular Applications

Visions: ***The traditional technology-driven transportation system is evolving from an era of providing simple transportation services into a more powerful data-driven intelligent era.***

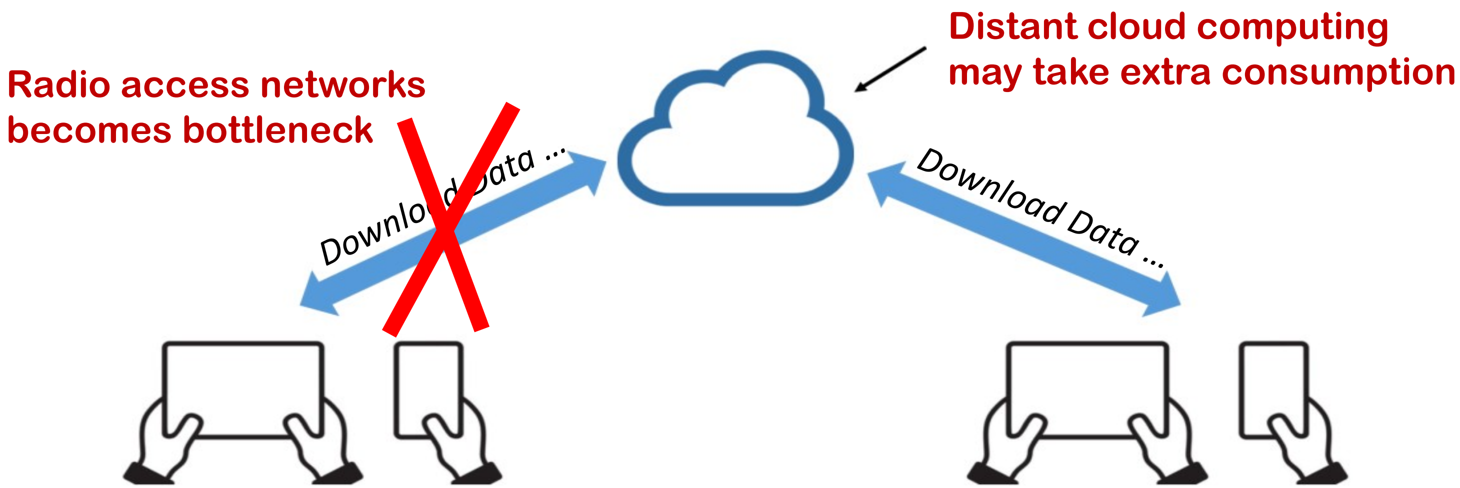


- **The vehicular applications require vehicles to access huge amount of Internet data.**
  - **Significant redundant traffic loads**
  - **limited channel bandwidth pose challenges for massive content delivery**
  - **Considerable delay for content delivery**

Especially for some delay-sensitive contents (e.g., video, image-aided navigation and live traffic information …)

From Cloud to Edge

- **Mobile Cloud Computing (MCC)**

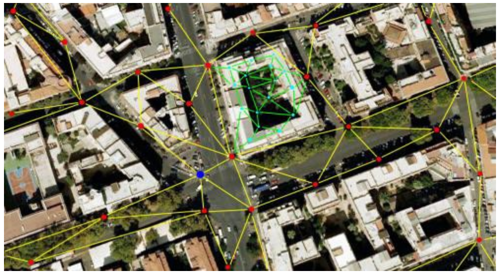


Some Limitations of Cloud Computing :

- ***Cloud servers are spatially far from vehicles -> The rapid increase in cost and latency***
- ***Cannot guarantee the tight Quality of Service (QoS) of delay-sensitive contents***
- ***The bottleneck of massive content delivery -> The utilization efficiency of the channel bandwidth is notably reaching its theoretical boundary.***

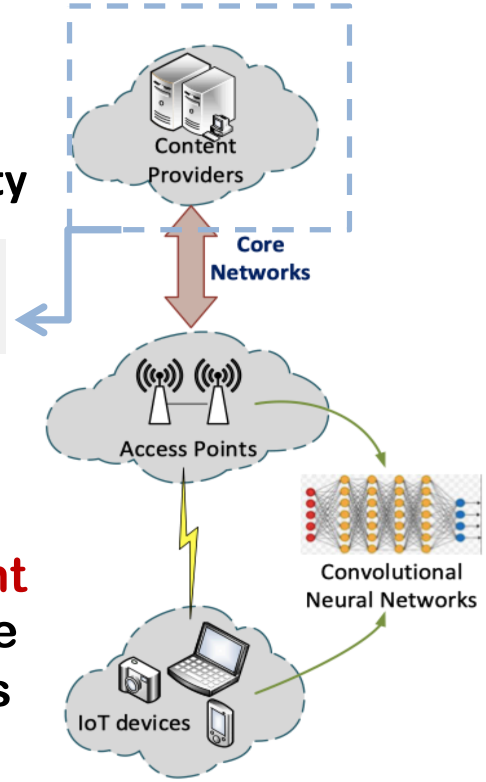
Edge Caching

- **Stemming from the studies：**



Content Delivery **Transmission Delay** **Privacy & Security**

***Discovery:* only a few contents are repeatedly downloaded upon request from vehicles, whilst the remaining large portion of the contents impose rather infrequent access demands.**



- **Edge Caching:**

  To **alleviate the redundant traffic** and **lower the content access latency** in IoVs, which caches contents in close proximity to vehicles by utilizing the storage resources at intermediate Roadside Units (RSUs).

Contents

Next Part

1 Introduction

2 **System Model**

3 Problem Formulation

4 Problem Solving

5 Performance Evaluation

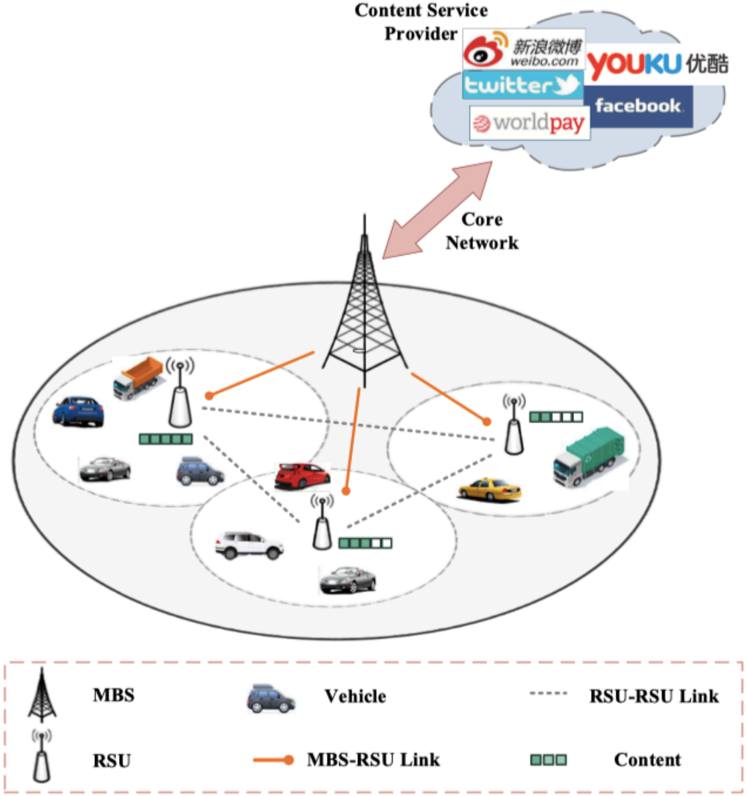
6 Conclusion

Network Architecture

- **A cooperative edge caching-supported IoV architecture with an MBS and N small cells, each of which with a RSU:**

Network Infrastructure:

- *An MBS connected with the service providers.*
- *N RSUs can be interoperable with each other and connected to the MBS.*
- *F available contents, s_f denote the size of content f.*
- *Each RSU is endowed with limited available cache capacity.*
- *Content popularity ρ_f obeys the Mandelbrot-Zipf distribution.*



Content Delivery Model

- ***Cooperative RSUs or MBS will incur certain costs for delivering the requested contents to the vehicles.***

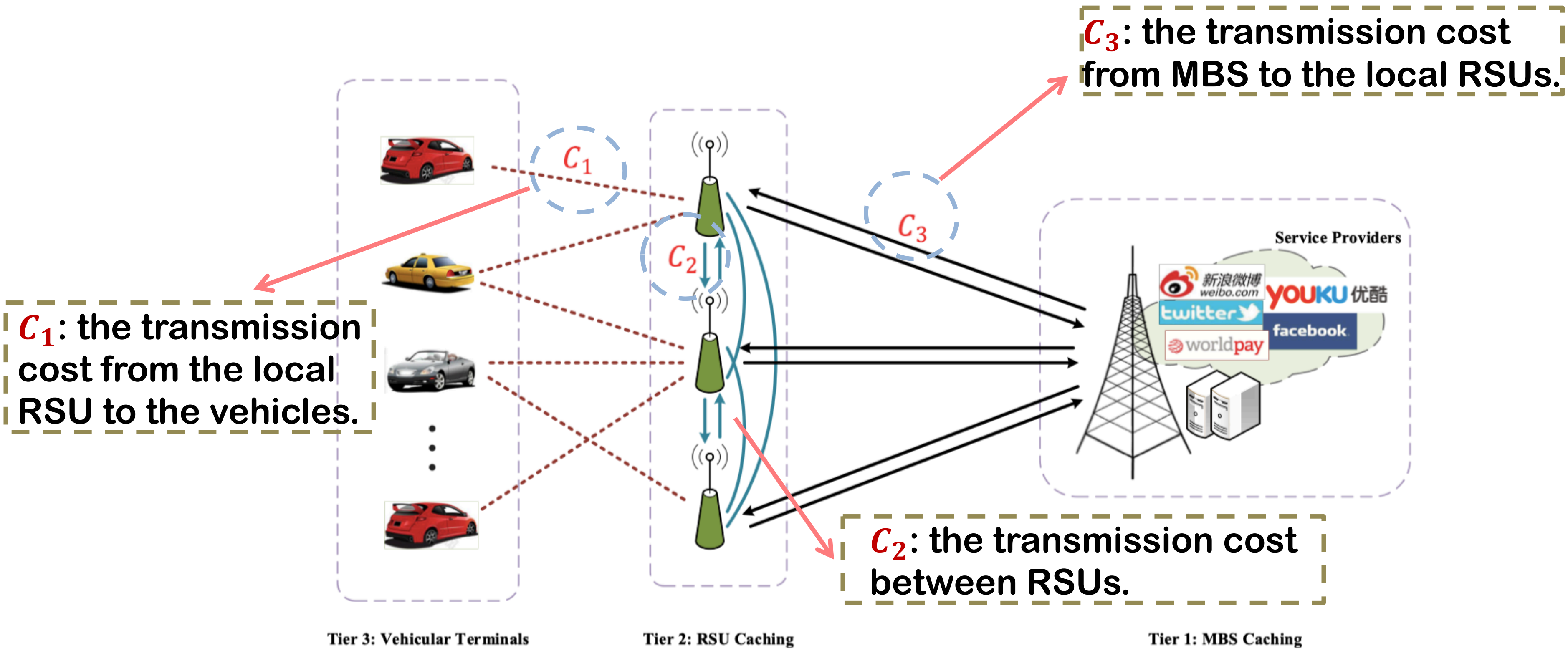


Fig. 2. Content Delivery Process in Hierarchical Networks.

Routing Decision & Cache Replacement

- ***The decision of any local RSU i for the requested content f can be represented by the binary decision variable***

$$a_{f,i,j}^{t} \in \{0,1\}, \qquad \forall j \in \mathcal{N} \cup \{N+1\} \begin{cases} a_{f,i,i}^{t} = 1 \\ a_{f,i,j}^{t} = 1 \ (j \in \mathcal{N} \backslash \{i\}) \\ a_{f,i,N+1}^{t} = 1 \end{cases}$$

➢ **Q:** ***Whether and which content should be replaced with the new ones when the cache capacity is fully occupied?***

➡

The content replacement control in RSU i:

$$\boldsymbol{c}_{f,i,i}^{t} = [\, c_{1,i,i}^{t}, c_{2,i,i}^{t}, \dots c_{F,i,i}^{t}]$$

Contents

Next Part

1 Introduction

2 Network Architecture

3 Problem Formulation

4 Problem Solving

5 Performance Evaluation

6 Conclusion

Vehicular Edge Network Architecture

- We aim to minimize the long-term overhead of content delivery in the system, the corresponding problem can be formulated as:

(P)

$$\min_{a_{f,i,j}} \lim_{T\to\infty} \frac{1}{T} \sum_{t=1}^{T} \sum_{f=1}^{F} \sum_{i=1}^{N} \sum_{j=1}^{N+1} \rho_f a_{f,i,j}^t C_{i,j}$$

Total overhead of ***content delivery***

$$\begin{aligned} s.\ t.\quad & C1: a_{f,i,j}^t \in \{0,1\}, \quad \forall f, \forall i, \forall j \\ & C2: a_{f,N+1,N+1}^t = 1, \quad \forall f \\ & C3: \sum_{j=1}^{N+1} a_{f,i,j}^t = 1, \quad \forall f, \forall i \\ & C4: \sum_{f=1}^{\mathcal{F}} a_{f,i,i}^t \cdot s_f \le R_i, \quad \forall i. \end{aligned}$$

the the integer nature of the binary caching decision

MBS has cached all available contents

content requested can only be handled by a RSU or MBS ultimately at one time slot

the limitation of RSU's cache capacity

Challenges

- **Challenges**
  - Only the local sub-optimal solution can be obtained in the system, not consider global optimal solution.
  - The caching decision variable is dynamically changing, and the objective function is NP-hard undoubtedly.
  - The feasible set of the problem is not convex and the complexity is very enormous.
- **Proposed solution**

- Extend the Markov Decision Process (MDP) to the multi-agent system.

→

- Formulate the process as a Markov Game (MG) model innovatively.

→

- Distributed MARL (Multi-Agent Reinforcement Learning) based edge caching method

State, Action and Reward Definition

- We approximate the cache replacement process in an available RSU as a **Markov Decision Process (MDP).**
  - *State*: *the current cache status* $a_{f,i,i}^{t} = \left(a_{1,i,i}^{t}, a_{2,i,i}^{t}, \dots a_{F,i,i}^{t}\right)$ *respect to the contents in a RSU; the arrived content requests* $q_{i,r}^{t}$ *in the current time slot t*
  - *Action*: *the decision* $a_{f,i,j}^{t}$ *of a RSU for the current requested contents; the content replacement control* $c_{f,i,i}^{t}$ *in a RSU in slot t.*
  - *Reward*: *To minimize the long-term overhead of content delivery in system, we use negative exponential function to transform the problem.*

$$\mathcal{R}\left(z_t^i, d_t^i\right) = e^{-\sum_f^{\mathcal{F}} \sum_j^{N+1} \rho_f a_{f,i,j}^{t} c_{i,j}}$$

Contents

Next Part

1 Introduction
2 Network Architecture
3 Problem Formulation
4 **Problem Solving**
5 Performance Evaluation
6 Conclusion

Reinforcement Learning : Q-Learning

- The iterative formula of Q-value of each-step can be obtained as follows:

Repeat until learning is stopped

$$\boxed{New\ Q(z_t, d_t)} = Q(z_t, d_t) + \varepsilon \underbrace{\left[r(z_t, d_t) + \varphi \max_{d_t} Q(z_{t+1}, d_{t+1}) - Q(z_t, d_t) \right]}_{\text{Bellman Equation}}$$

- $Q(z_t, d_t)$: *the Q value of admissible action d_t under the state z_t*
- $\varepsilon \in (0, 1)$: *learning rate parameter*
- z_{t+1} : *the new state*
- $\max_{d_t} Q(z_{t+1}, d_{t+1})$: *estimate of optimal future value*
- $r(z_t, d_t)$: *immediate reward after executing an admissible action d_t*
- *Discounting factor φ : indicate the importance of the predicted future rewards*

Markov Game Model

- Conventional RL has not considered **the influence of environment by other agents** when a certain agent interacts separately.
- Extend the MDP to a multi-agent system, and further formulate the problem as a **Markov (a.k.a. Stochastic) Game (MG) model**.

Definition of Markov Game

$$\{N,\ \mathcal{Z}, D_1,\ \cdots,\ D_N, p, r_1,\ \cdots, r_N,\ \gamma\}$$

Consider joint action

Avoid the bias in individual decision

- N: ***the number of agents***
- S: ***system state space***
- $D_i\ (i = 1,\ \cdots,\ N)$: ***a discrete action space of i-th agent, the joint action space of all agents can be represented as*** $\mathcal{D} = D_1 \times \cdots \times D_N$
- $p: \mathcal{Z} \times \mathcal{D} \times \mathcal{Z} \rightarrow [0, 1]$: ***the state transition probability map, the state make the transition based on a probability*** $p_{z_t z_{t+1}}(d_t^1, \cdots d_t^N)$
- $r_i: \mathcal{Z} \times \mathcal{D} \rightarrow \mathbb{R}$: ***the reward function for agent i***

Distributed MARL based Method

- **Nash equilibrium strategy: an optimal joint strategy of the system**

$$(\pi_1^*, \cdots, \pi_i^*, \cdots, \pi_N^*)$$

Each agent's strategy π_i^* is the best response to the others'

- **Any agent cannot achieve a higher reward by changing to any other strategy**

$$V_i(s, \pi_1^*, \cdots, \pi_i^*, \cdots, \pi_N^*) \geq V_i(s, \pi_1^*, \cdots, \pi_i, \cdots, \pi_N^*) \qquad \forall \pi_i \in \Pi_i$$

- **At time slot t, each agent executes its action under the current state. After that, it observes its own immediate reward, all other agents' actions and rewards, as well as the new state.**

maintains a model of other agents' Q-values performs updates over its own Q-values simultaneously.

Distributed MARL based Method

- **Redefine Q-function:** $Q_i(z_t, d_t^1, \cdots, d_t^N)$

when all agents follow a specified joint Nash equilibrium strategy

Nash Q-function for agent i:
$Q_i^*(z_t, d_t^1, \cdots, d_t^N)$

- **The stage games** $\left(Q_t^1(z_{t+1}),\ \cdots, Q_t^i(z_{t+1}), \cdots, Q_t^N(z_{t+1})\right)$ **will be formed with the Nash equilibrium strategies** $\pi_1^*(z_{t+1}), \cdots, \pi_i^*(z_{t+1}), \cdots, \pi_N^*(z_{t+1})$ **under certain restrictions.**

- **The iterative formula of Q-value of agent i in each time slot can be obtained as:**

Agent can derive the Nash equilibrium and choose its actions accordingly based on learned Q-values.

$$
\begin{aligned}
&Q_{t+1}^i\left(z_t, d_t^i, \cdots, d_t^N\right) \\
&= (1-\beta_t)\cdot Q_t^i\left(z_t, d_t^1, \cdots, d_t^N\right) + \beta_t\left[r_t^i + \gamma \,\mathrm{Nash}\, Q_t^i(z_{t+1})\right]
\end{aligned}
$$

Contents

Next Part

1 Introduction

2 Network Architecture

3 Problem Formulation

4 Problem Solving

5 **Performance Evaluation**

6 Conclusion

Performance Evaluation

For performance comparison, we introduce the following three benchmark algorithms:

- ***Independent RL (IRL):* *Each agent learns and makes decision separately through the repeated interaction, without considering the influence of other agents.***
- ***Least Frequently Used (LFU):* *When the cache capacity of each RSU is full, replace the content with the least requested times firstly.***
- ***Least Recently Used (LRU): When the cache capacity of each RSU is full, replace the least recently used content firstly.***

The convergence performance:

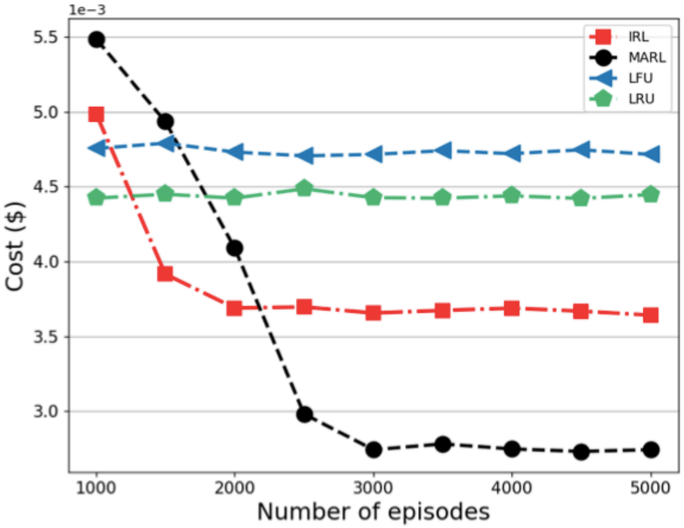
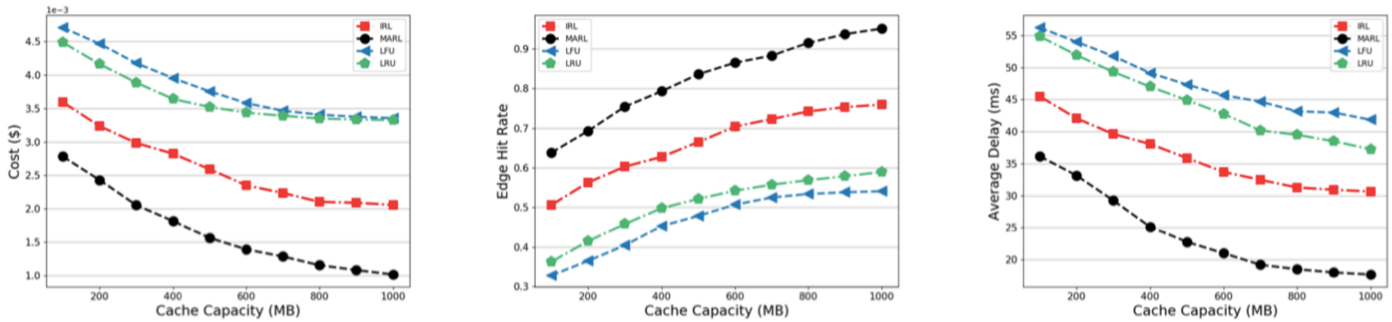


Fig. 3. Content access cost versus the number of epsiodes.

Performance Evaluation

Compare the performance metrics:
①The total access cost ②Edge hit rate ③Average delay

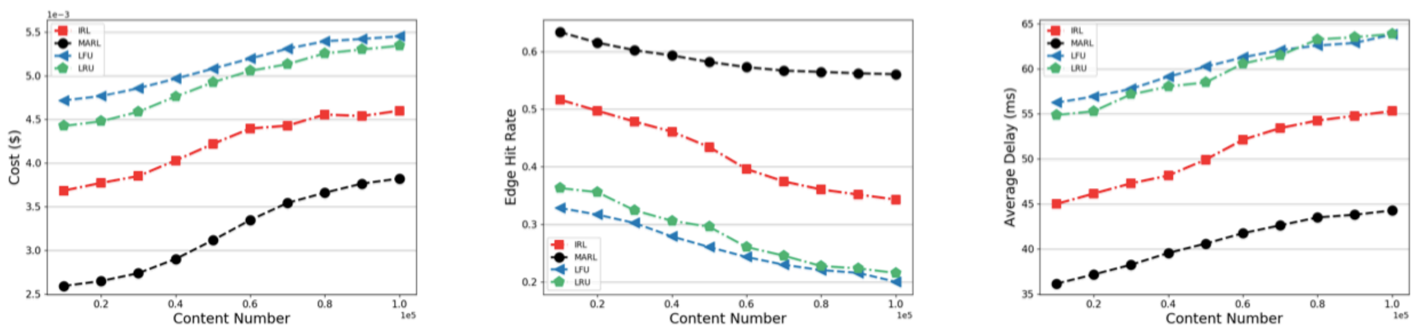
The impact of cache capability of RSUs:



(a) (b) (c)

Fig. 4. (a) Content access cost versus the cache capacity of RSUs. (b) Edge hit rate versus the cache capacity of RSUs. (c) Average Delay versus the cache capacity of RSUs.

The impact of the number of contents in the system:



(a) (b) (c)

Fig. 5. (a) Content access cost versus the number of contents. (b) Edge hit rate versus the number of contents. (c) Average Delay versus the number of contents.

Contents

Next Part

1 Introduction

2 Network Architecture

3 Problem Formulation

4 Problem Solving

5 Performance Evaluation

6 Conclusion

Conclusion

- ❑ Presented a cooperative edge caching architecture for IoVs.
- ❑ Formulated the optimization problem to minimize the long-term overhead of content delivery.
- ❑ Extended the MDP to the context of multi-agent system.
- ❑ Formulated the process as a Markov Game (MG) model.
- ❑ A distributed MARL based edge caching method.

- **Future work**

✓ The exponential increase of state-action space

–Improvement , Deep Neural Networks, …

✓ Exploit a more complex system

–User mobility, Vehicle-to-Vehicle caching, …

✓……

Q&A

Q & A

Reporter : Kai Jiang
Email : jiangkai0112@gmail.com