Modeling the Collaborative Edge Data Caching Problem via a Dynamic DCOP

Extended Abstract

Ziyang Song
College of Computer Science,
Chongqing University
Chongqing, China
202314131079T@stu.cqu.edu.cn

Ziyu Chen*
College of Computer Science,
Chongqing University
Chongqing, China
chenziyu@cqu.edu.cn

Jinhui Huang
College of Computer Science,
Chongqing University
Chongqing, China
huangjinhuicqu@gmail.com

Cheng Zhang
College of Computer Science,
Chongqing University
Chongqing, China
bootan@cqu.edu.cn

Jingyuan He
College of Computer Science,
Chongqing University
Chongqing, China
ibm_hjy@cqu.edu.cn

ABSTRACT

The Collaborative Edge Data Caching (CEDC) problem poses a significant challenge in Mobile Edge Computing (MEC). It's a research focus to address the problem from the service providers' perspective that requires the optimal caching strategy for service providers to maximize their caching revenue, subject to capacity and latency constraints. However, current research primarily focuses on centralized methods, neglecting the distributed and dynamic nature of CEDC. Accordingly, we first propose to use a Dynamic Distributed Constraint Optimization Problem (D-DCOP) to model the problem in a distributed manner, where capacity, latency constraints and caching revenue are dynamically mapped into local hard constraints and constraint utilities between edge servers according to changes in user requests. The proposed model enables each edge server to make its caching strategy through information exchange with neighboring edge servers. We further present a local search framework for CEDC to handle local hard constraints in the model and apply it to two classic local search algorithms, DSA and MGM, along with specific modifications to avoid repetitive computation. We empirically confirm the superiority of our distributed model and algorithms over state-of-the-art centralized solvers for CEDC.

KEYWORDS

Dynamic DCOP, collaborative edge data caching, local search algorithms, mobile edge computing

ACM Reference Format:
Ziyang Song, Ziyu Chen, Jinhui Huang, Cheng Zhang, and Jingyuan He. 2025. Modeling the Collaborative Edge Data Caching Problem via a Dynamic DCOP: Extended Abstract. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 3 pages.

*Corresponding author

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

1 INTRODUCTION

The collaborative edge data caching (CEDC) problem is a significant challenge in MEC [1, 2], focusing on how to efficiently store and access frequently used data in edge environments [3]. Most existing studies for the CEDC problem concentrate on optimizing overall caching performance [4], e.g. minimizing user data retrieval latency [5], maximizing cache hit rate [6], minimizing system energy consumption [7, 8]. However, these studies consider the problem from the perspective of mobile network operators, overlooking the needs and concerns of service providers [9]. Unlike mobile network operators, service providers consider not only the reduction of user data retrieval latency but also the optimization of caching costs on edge servers. Xia et al. [9] were one of the earliest to model this problem as a constrained optimization problem, proposing a Lyapunov optimization based algorithm aimed at minimizing caching cost. Subsequently, Liu et al. [10] formulated the problem as an integer programming problem by introducing caching benefit and proposed an exact algorithm (IPEDC) for it. Since the CEDC problem has been proven to be $\mathcal{NP}$-hard, Xia et al. [11] developed an approximation algorithm (AEDC) to maximize caching revenue. Chen et al. [12] formulated the problem as a constrained optimization problem to reduce retrieval latency within the service provider's budget. Xia et al. [13] were the first to consider the dynamic nature of the problem, proposing a system model aimed at maximizing caching revenue and introducing the OL-MEDC algorithm.

However, the MEC system is a typical distributed scenario [14], where computing and storage resources are deployed on edge servers. The above studies are all based on a centralized model. As the scale of the problem increases, these centralized approaches place huge pressure on network bandwidth, leading to delays in caching strategy transmission and insufficient scalability. Therefore, there is an urgent need for dynamic distributed solvers for the CEDC problem to effectively adapt to this distributed environment.

In the paper, we formulate the CEDC problem as a D-DCOP [15] to accommodate its distributed and dynamic nature and accordingly present the distributed algorithms for it. To the best of our knowledge, this work is the first attempt to solve the CEDC problem from the service providers' perspective in a distributed manner.

2 A CEDC MODEL BASED ON D-DCOP

We adopt a D-DCOP to reconstruct the CEDC model as described in [13]. Specifically, we formulate the CEDC problem over a period of time T as a sequence of DCOPs: $\mathcal{D}_1, \ldots, \mathcal{D}_T$. Each $\mathcal{D}_t = \langle S^t, \lambda^t, D^t, F^t \rangle$ is defined as follows.

• $S^t = \{S_1, \ldots, S_n\}$ is a set of agent. Each agent S_i represents an edge server in the CEDC problem.

• $\lambda^t = \{\lambda_1^t, \ldots, \lambda_n^t\}$ is a set of variables, where λ_i^t is controlled solely by agent S_i and represents the caching strategy of S_i in time slot t. Each $\lambda_i^t = \{\lambda_{i,1}^t, \ldots, \lambda_{i,z}^t\}$, where $\lambda_{i,k}^t$ indicates whether agent S_i chooses to cache $data_k$ in time slot t, and $|\lambda_i^t| = |Data|$.

• $D^t = \{D_1^t, \ldots, D_n^t\}$ is a set of variable domains. Each $D_i^t \in D^t$ consists of a set of finite allowable values for variable $\lambda_i^t \in \lambda^t$. For each $\lambda_{i,k}^t \in \lambda_i^t$, its domain is $\{0, 1\}$, where 0 indicates that agent S_i does not cache $data_k$ in time slot t, and 1 indicates that it does.

• F^t is a set of constraint utility functions, where $f_{ij}^t \in F^t : D_i \times D_j \to \mathbb{R}_{\geqslant 0}$ specifies the nonnegative utility for each value combination of λ_i^t and λ_j^t subject to $\Phi_{i,j} \leq \Phi_{\max}$. Here, $\Phi_{i,j}$ and $\Phi_{\max}$ denote hop count between edge servers S_i and S_j, and latency limit [16]. Two local hard constraints and f_{ij}^t are defined as follows.

(a) The *local capacity constraint* and *local latency constraint* are defined by Eq. (1) and Eq. (2), respectively. Here, $|data_k|$ denotes the size of $data_k$, $cache_{max}$ represents the cache limit [17], $N(S_i)$ is the set of agents with hops not exceeding $\Phi_{\max}$ from S_i, and $\tau_{q,k}^t$ indicates whether user u_q requests $data_k$ in time slot t.

$$\sum_{data_k \in Data} \lambda_{i,k}^t \cdot |data_k| \leq cache_{\max}, \forall t \in T \tag{1}$$

$$\sum_{S_l \in N(S_i) \cup \{S_i\}} \lambda_{l,k}^t \neq 0, \quad s.t. : \exists u_q \in U_i, \tau_{q,k}^t = 1 \tag{2}$$

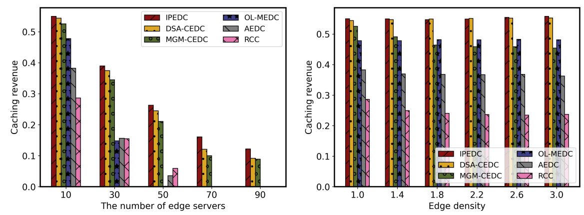
(b) The *constraint utility* f_{ij}^t, defined by Eq. (3), incorporates two soft constraints: the *caching cost gain* BR_{ij}^t, which quantifies the difference between the cost of transmitting $data_k$ from the cloud to S_i and S_j and the actual cost of storing $data_k$ in these servers, and the *data caching benefit* BD_{ij}^t, which represents the total caching benefit derived from data retrieval by users $u_q \in U_i^t \cup U_j^t$. Here, γ denotes the service provider's priority for lowering user data retrieval latency. Accordingly, we formulate the CEDC problem over a period of time T as Eq. (4).

$$f_{ij}^t \left(\lambda_i^t, \lambda_j^t\right) = \gamma \cdot BD_{ij}^t \left(\lambda_i^t, \lambda_j^t\right) + BR_{ij}^t \left(\lambda_i^t, \lambda_j^t\right) \tag{3}$$

$$\max \lim_{T \to \infty} \sum_{t=1}^{T} \sum_{f_{ij}^t \in F^t} f_{ij}^t(\lambda_i^t, \lambda_j^t), \quad s.t. : Eq.\ (1), Eq.\ (2) \tag{4}$$

3 LOCAL SEARCH FRAMEWORK FOR CEDC

We propose a local search framework to address the dynamics and local hard constraints in the model. To tackle changes in user data requests over time, each agent S_i updates its *localUserRequireDataView* (a set of data requests from users covered by all $S_l \in \{S_i \cup N(S_i)\}$ in time slot t) by exchanging userRequireDataMessage with $S_j \in N(S_i)$ and initializes λ_i^t to λ_i^{t-1} at the beginning of time slot t. To tackle hard constraints, each agent S_i exchanges initMessage and suggestMessage to update *localDataView* (a set of caching strategies for all S_l in time slot t), *localDataDelayView* (a set of latencies for all S_l to retrieve data in time slot t) and *acceptInitMap* (a set of variables indicating whether all $S_j \in N(S_i)$ satisfy local hard constraints). When all variables in *acceptInitMap* are set to 1, it indicates that all S_l have satisfied the *local capacity constraint* and *local latency constraint*, marking the completion of the hard constraint satisfaction phase. Through this framework, each agent S_i provides a feasible assignment and latency information regarding $N(S_i)$ for local search algorithm for CEDC in each time slot.



(a) The number of edge servers **(b) Edge density**

Figure 1: Caching Revenue Under Different Settings

4 DSA-CEDC AND MGM-CEDC

Based on the framework presented in Section 3, we introduce specific modifications to DSA [18] and MGM [19], and propose two distributed algorithms for CEDC, named DSA-CEDC and MGM-CEDC, respectively. The specific modifications encompass: (1) we modify the computation for the sum of f_{ij}^t to avoid double-counting of $BR_{ij}^t(\lambda_i^t, \lambda_j^t)$; (2) We further refine the feasible D_i^t by rechecking the *local capacity constraint* and *local latency constraint*. (3) We update *localDataDelayView* by exchanging updateDataDelayMessage among agents, ensuring accurate and up-to-date latency information. These specific modifications can be easily applied to any local search algorithm.

5 EXPERIMENTAL EVALUATION

We evaluate DSA-CEDC and MGM-CEDC using a real-world EUA dataset [20]. Due to space constraints, we present only the results with a brief summary. Detailed information on the experimental setup and configuration is available in [10, 11, 13].

In Fig. 1(a), as the number of edge servers increases, the expanded cache space enhances the chances of data transmissions but also raises caching costs, ultimately reducing caching revenue. In scenarios with more servers, OL-MEDC, AEDC, and RCC fail to generate positive caching revenue, whereas DSA-CEDC and MGM-CEDC achieve 75.2% and 72.9% of IPEDC's revenue, respectively. In Fig. 1(b), as the network density reaches 3.0, MGM-CEDC's average caching revenue drops by 13.5% due to its additional round of message passing compared to DSA-CEDC. Higher network density increases communication load and coordination complexity, causing greater performance fluctuations in MGM-CEDC. In contrast, DSA-CEDC's lower communication overhead ensures stable performance across all network densities. These experiments indicate that DSA-CEDC and MGM-CEDC closely approximate the exact algorithm IPEDC, demonstrating their effectiveness in generating caching strategies even in more complex and large-scale scenarios.

REFERENCES

[1] Yuyi Mao, Changsheng You, Jun Zhang, Kaibin Huang, and Khaled B Letaief. 2017. A survey on mobile edge computing: The communication perspective. *IEEE communications surveys & tutorials* 19, 4 (2017), 2322–2358.

[2] Nasir Abbas, Yan Zhang, Amir Taherkordi, and Tor Skeie. 2017. Mobile edge computing: A survey. *IEEE Internet of Things Journal* 5, 1 (2017), 450–465.

[3] Ge Wang, Chunlin Li, Yong Huang, Xiangli Wang, and Youlong Luo. 2022. Smart contract-based caching and data transaction optimization in mobile edge computing. *Knowledge-Based Systems* 252 (2022), 109344.

[4] Yongseok Oh, Jongmoo Choi, Donghee Lee, and Sam H Noh. 2012. Caching less for better performance: balancing cache size and update cost of flash memory cache in hybrid storage systems.. In *FAST*, Vol. 12.

[5] Utsav Drolia, Katherine Guo, Jiaqi Tan, Rajeev Gandhi, and Priya Narasimhan. 2017. Cachier: Edge-caching for recognition applications. In *2017 IEEE 37th international conference on distributed computing systems (ICDCS)*. IEEE, 276–286.

[6] Peng Shu and Qingwei Du. 2020. Group behavior-based collaborative caching for mobile edge computing. In *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Vol. 1. IEEE, 2441–2447.

[7] Guiyang Luo, Haibo Zhou, Nan Cheng, Quan Yuan, Jinglin Li, Fangchun Yang, and Xuemin Shen. 2019. Software-defined cooperative data sharing in edge computing assisted 5G-VANET. *IEEE Transactions on Mobile Computing* 20, 3 (2019), 1212–1229.

[8] Xiaofei Wang, Chenyang Wang, Xiuhua Li, Victor CM Leung, and Tarik Taleb. 2020. Federated deep reinforcement learning for Internet of Things with decentralized cooperative edge caching. *IEEE Internet of Things Journal* 7, 10 (2020), 9441–9455.

[9] Xiaoyu Xia, Feifei Chen, Qiang He, John Grundy, Mohamed Abdelrazek, and Hai Jin. 2021. Online Collaborative Data Caching in Edge Computing. *IEEE Transactions on Parallel and Distributed Systems* 32, 2 (2021), 281–294.

[10] Ying Liu, Qiang He, Dequan Zheng, Xiaoyu Xia, Feifei Chen, and Bin Zhang. 2020. Data caching optimization in the edge computing environment. *IEEE transactions on services computing* 15, 4 (2020), 2074–2085.

[11] Xiaoyu Xia, Feifei Chen, Qiang He, Guangming Cui, Phu Lai, Mohamed Abdelrazek, John Grundy, and Hai Jin. 2020. Graph-based data caching optimization for edge computing. *Future generation computer systems* 113 (2020), 228–239.

[12] Xiaoyu Xia, Feifei Chen, Guangming Cui, Mohamed Abdelrazek, John Grundy, Hai Jin, and Qiang He. 2020. Budgeted data caching based on k-median in mobile edge computing. In *2020 IEEE International Conference on Web Services (ICWS)*. IEEE, 197–206.

[13] Xiaoyu Xia, Feifei Chen, Qiang He, Guangming Cui, John Grundy, Mohamed Abdelrazek, Athman Bouguettaya, and Hai Jin. 2021. OL-MEDC: An online approach for cost-effective data caching in mobile edge computing systems. *IEEE Transactions on Mobile Computing* 22, 3 (2021), 1646–1658.

[14] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge computing: Vision and challenges. *IEEE internet of things journal* 3, 5 (2016), 637–646.

[15] Khoi D Hoang and William Yeoh. 2022. Dynamic continuous distributed constraint optimization problems. In *International Conference on Principles and Practice of Multi-Agent Systems*. Springer, 475–491.

[16] Ruikun Luo, Hai Jin, Qiang He, Song Wu, and Xiaoyu Xia. 2023. Enabling balanced data deduplication in mobile edge computing. *IEEE Transactions on Parallel and Distributed Systems* 34, 5 (2023), 1420–1431.

[17] Shuiguang Deng, Zhengzhe Xiang, Javid Taheri, Mohammad Ali Khoshkholghi, Jianwei Yin, Albert Y. Zomaya, and Schahram Dustdar. 2021. Optimal Application Deployment in Resource Constrained Distributed Edges. *IEEE Transactions on Mobile Computing* 20, 5 (2021), 1907–1923.

[18] Weixiong Zhang, Guandong Wang, Zhao Xing, and Lars Wittenburg. 2005. Distributed stochastic search and distributed breakout: properties, comparison and applications to constraint optimization problems in sensor networks. *Artificial Intelligence* 161, 1-2 (2005), 55–87.

[19] Rajiv T Maheswaran, Jonathan P Pearce, Milind Tambe, et al. 2004. Distributed Algorithms for DCOP: A Graphical-Game-Based Approach.. In *PDCS*. Citeseer, 432–439.

[20] Phu Lai, Qiang He, Mohamed Abdelrazek, Feifei Chen, John Hosking, John Grundy, and Yun Yang. 2018. Optimal edge user allocation in edge computing with variable sized vector bin packing. In *Service-Oriented Computing: 16th International Conference, ICSOC 2018, Hangzhou, China, November 12-15, 2018, Proceedings 16*. Springer, 230–245.