Collaborative Edge Caching with Multiple Virtual Reality Service Providers Using Coalition Games

Chun-Che Lin, Yao Chiang, Hung-Yu Wei
Department of Electrical Engineering, National Taiwan University, Taiwan
Taipei, Taiwan
Email address: hywei@ntu.edu.tw

***Abstract*—Mobile edge computing (MEC) and 5G networks can provide ultra-low latency connections. Combining the two, caching services at the network edge can greatly reduce the delay of virtual reality (VR) and augmented reality (AR) services, enhancing the Quality of Service (QoS) for users. In this paper, we investigate an efficient collaborative service caching scheme between multiple service providers (SPs) with a game-theoretical approach. We model SPs as players who care about nothing but their profits and can form coalitions by sharing edge server resources as well as costs with other members. More than one coalition can be formed in an edge server. Our algorithm guarantees to reach a Nash equilibrium, where no one has the incentive to deviate. Simulation results show that through the proposed collaboration scheme, SPs can reach a higher profit compared to several baselines as well as previously proposed schemes.**

***Index Terms*—Service caching, edge computing, coalition game, game theory, resource allocation**

I. Introduction

During the coronavirus (COVID-19) pandemic, where activities can only be held virtually, the future of virtual reality (VR) and augmented reality (AR) only becomes brighter as they can not only provide a more immersive virtual conference or meeting experience, but also give treatments or supports for patients suffering from psychological disorders as well as physical illness [1]. However, with the processing of computation-intensive VR/AR services combined with the heavy traffic of core networks, it has become quite a challenge for VR/AR services deployed on the cloud to satisfy the Quality of Services (QoS) requirements of users.

With the development of multi-access edge computing (MEC) [2] and 5G technology [3], 5G-enabled edge servers can be exploited to improve users' QoS. By offloading tasks or caching services to the network edge, in the proximity of users, latency and therefore users' QoS can be greatly improved [4] [5] [6]. To increase the cache hit ratio, collaborative service caching among servers is proposed [7] [8] [9]. Game-theoretic models have also been suggested to examine the potential for collaborations between different service providers (SPs) [10] [11]. However, the possibility of having more than one coalition in each edge server for more efficient collaborative service caching has never been considered in past studies.

In this paper, we investigate how SPs can maximize their profits by forming multiple coalitions and sharing VM resources within a single edge server. We first formulate a coalition game for SPs to cache their services to edge servers to reduce latency and freely collaborate with other SPs in the same edge server to share the costs of VM resources. Later, a distributed algorithm is proposed to find the Nash equilibrium, where none of the SPs have the incentive to switch to another coalition without violating the Pareto criterion.

The remainder of this paper is organized as follows. The system model is introduced in section II. The coalition game formation of the collaborative service caching scheme is formulated in section III. A distributed algorithm is proposed in section IV to reach the Nash equilibrium. Simulation results are presented in section V. Finally, conclusions are drawn in section VI.

II. System Model

In this section, we will first introduce the coalition game-based system model for service caching. Secondly, the mathematical model of the delay, the revenue, the cost, and the profit of SPs will be discussed.

The system model is shown in Fig. 1. There are four key components, including 1 cloud server, M edge servers, N SPs, and their respective users. The set of servers is $\mathcal{M} = \{0, 1, \cdots, M\}$, where 0 denotes the cloud server and $1 \sim M$ denotes the edge servers, which follow the P1935-standard [12]. The set of SPs is $\mathcal{N} = \{1, \cdots, N\}$, presented in the blue box in Fig. 1. Each of them has a set of users. We assume the switching costs to be high and the users won't switch to other SPs in the short term [13]. Edge servers are provided by infrastructure providers, and SPs can lease resources from them. A coalition is formed when SPs leasing VMs from the same edge server decide to share their resources as well as the costs with each other. Each SP can either has its service directly from the cloud server or caches its service to an edge server to reduce delay. If an SP decides to use an edge server, it can also decide whether to lease and pay for the VMs in the edge server provided by the infrastructure providers, or share the VMs with other SPs using the same edge server to reduce cost and utilize idle resources.

For example, in Fig. 1, the rightmost SP has its service straight from the cloud server, making it unable to meet the delay requirement of its user, and it ends up not getting paid. On the other hand, the second rightmost SP caches its service to an edge server to reduce delay, but doesn't share its VM with other SPs. As a result, it gets paid by its users for meeting

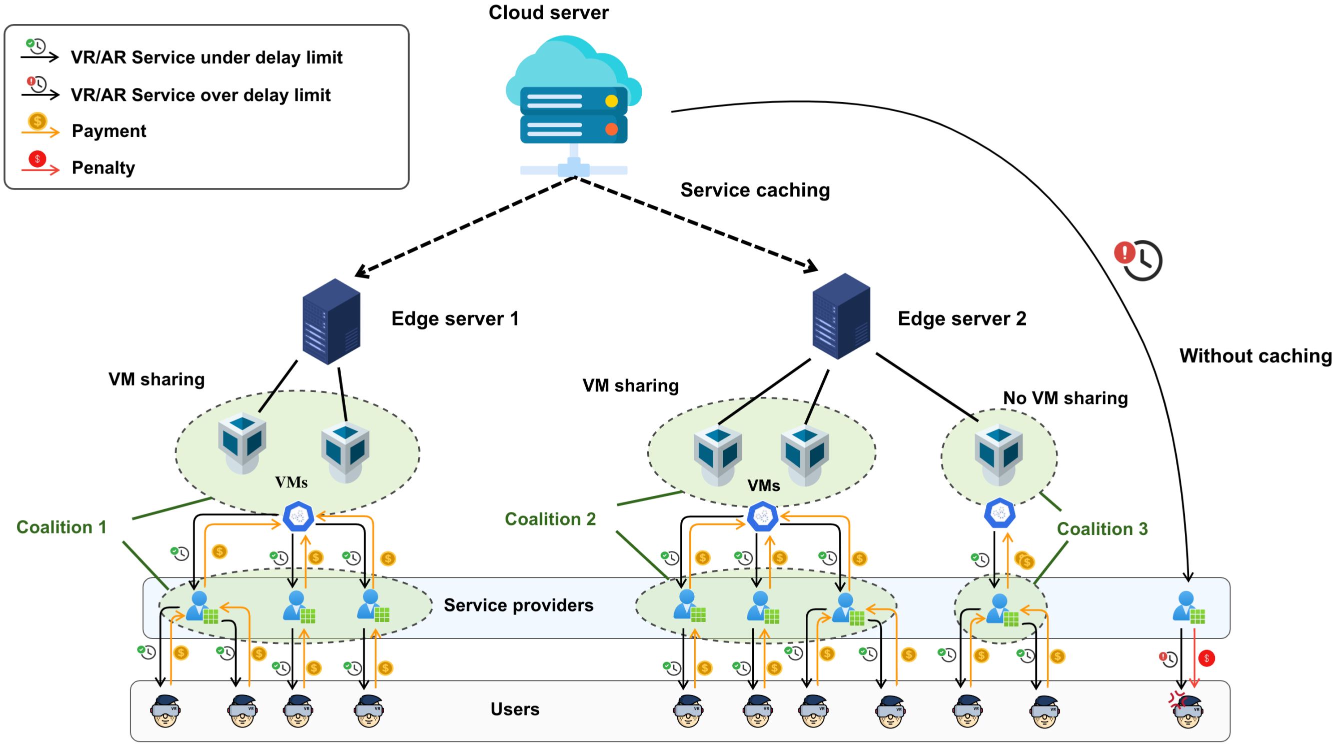

Fig. 1. System model.

the delay requirement, while also pays for the whole cost of its leased VM resource. The rest of the SPs, however, not only cache their services to edge servers, but also form coalitions to share VMs, resulting in lower costs and higher revenues.

A. Delay Model

The delay of the service of each SP consists of propagation delay and response delay. When an SP is served by the cloud, the response delay would be a constant. But when it is served by a coalition in an edge server, the response delay would be the total response time of an M/M/c queue. The delay formula is formulated as

$$D_n = D_{p,n} + D_{r,k} \tag{1}$$

where

$$D_{p,n} = \alpha d_n \tag{2}$$

$$D_{r,k} = \begin{cases} \dfrac{E(\text{vn}_k, \frac{\text{vn}_k \cdot \mu}{\text{uns}_k \cdot \lambda})}{\text{vn}_k \cdot \mu - \text{uns}_k \cdot \lambda} + \dfrac{1}{\mu}, & \text{if } 1 \leq m \leq M \\ \text{rd}, & \text{if } m = 0 \end{cases} \tag{3}$$

where

$$E(g, \nu) = \frac{1}{1 + (1-\nu)\left(\frac{g!}{(gp)^g}\right)\sum_{l=0}^{g-1}\frac{(gp)^l}{l!}} \tag{4}$$

where D_n is the total delay of the service of SP n, k is the coalition SP n is in, m is the server coalition k is in, $D_{p,n}$ is the propagation delay from the server of SP n to itself, $D_{r,k}$ is the total response delay of coalition k, α is the ratio of propagation delay to distance traveled, d_i is the distance between SP n and its server, E is Erlang's C formula [14], the probability of a job in an M/M/c queue to be sent into the queue as opposed to being served immediately, rd is the response delay of a cloud server, vn_k is the number of VMs in coalition k, uns_k is the total number of users served by the SPs in coalition k, μ is the processing rate of each VM, and λ is the request rate of each user.

B. Revenue Model

The revenue of each SP comes from the payments of its users. Each SP provides various levels of delay guarantee, each with a different price as well as penalty. The user will pay its SP if the delay of the service it received is under the agreed delay guarantees. On the contrary, if the delay requirement is not met, the SP will have to pay the users as penalties.

The revenue of SP n can be formulated as

$$R_n = \sum_{j \in S_n} q_j \tag{5}$$

where

$$q_j = \begin{cases} \rho_j, & \text{if } D_n \leq T_j^* \\ -\gamma\rho_j, & \text{otherwise} \end{cases} \tag{6}$$

where

$$\rho_j = \mathrm{lp} \cdot l_j \tag{7}$$

$$T_j^* = T - \mathrm{ld} \cdot (l_j - 1) \tag{8}$$

where R_n is the revenue of SP n, S_n is the set of users served by SP n, q_j is the money user j has to pay to its SP, ρ_j is the service subscription price of user j, $0 < \gamma \leq 1$ is the penalty to price ratio, D_n is the delay of the service of SP n defined in (1), T_j^* is the delay requirement of user j, $l_j \in \mathbb{N}$ is the subscription level of user j, lp is the price of each level, ld is the delay difference of each level, and T is the delay of the base subscription level i.e. level 1.

C. Cost Model

The cost of each SP is the payment to infrastructure providers for leasing edge server VMs. The formula is written as

$$C_n = \begin{cases} \mathrm{vn}_k \mathrm{vc} \frac{\mathrm{unp}_i}{\mathrm{uns}_k}, & \text{if } 1 \leq m \leq M \\ 0, & \text{if } m = 0 \end{cases} \tag{9}$$

where C_n is the cost of SP n, m is the server SP n uses, k is the coalition SP n is in, vn_k is the number of VMs in coalition k, vc is the cost of a VM, uns_k is the total number of users served by the SPs in coalition k, and unp_i is the number of users served by SP n. If an SP doesn't share VMs with others, then there will be only one SP in its coalition, it would therefore have to pay for all of the VMs it leases. If an SP doesn't cache its service to an edge server i.e. $m = 0$, then the cost is 0 as it doesn't have to pay for edge server resources.

D. Profit Model

The profit of each SP is simply the revenue minus the cost, which can be formulated as

$$P_n = R_n - C_n \tag{10}$$

where P_n is the profit of SP n, R_n is the revenue of SP n defined in (5), and C_n is the cost of SP n defined in (9).

E. Problem Formulation

The optimization goal is to maximize the total profit of the system under the resource limit of each edge server, which can be formulated as

$$\begin{aligned} &\max \sum_{n=1}^{N} P_n \\ &\text{s.t.} \sum_{j \in V_m} \mathrm{cpu}_j \leq \mathrm{cl}_m, \forall m \in [1, M] \end{aligned} \tag{11}$$

where P_n is the profit of SP n defined in (10), V_m is the set of VMs of edge server m, cpu_j is the CPU unit of VM j, and cl_m is the CPU unit limit of edge server m.

III. Collaborative service caching scheme

In this section, we will discuss the properties of the collaborative service caching scheme.

In order to investigate the collaboration between SPs, a coalition game is introduced, the coalition game is formulated as follows:

- Servers: There are 1 cloud server and M edge servers
- Players: There are N SPs, and they are the players.
- Strategies: Each player has 3 strategies.
  1) Stay on the cloud server.
  2) Cache its service by leasing VMs on an edge server.
  3) Cache its service by joining a coalition on an edge server to share VMs with other members in the same coalition.
- Utility: The utility of a player is the profit of the player, defined in (10).

A coalition is a group of players sharing the same set of VMs and their costs in an edge server. It can consist of one or more players. There can be multiple coalitions inside an edge server. Players can freely switch between different coalitions on different edge servers to improve their utilities on the condition of not reducing other players' utilities.

The utility of an SP is dependent on its own as well as other SPs' strategies, denoted as $u_n(a_n, a_{-n})$, where a_n is the strategy of SP n i.e. the server and coalition it selects, while $a_{-n} = \{a_1, \cdots, a_{n-1}, a_{n+1}, \cdots, a_N\}$ is the strategies of other SPs. $\mathcal{F}_{m,k}$ denotes the coalition k inside the server m.

Based on the coalition formation game model [15], we introduce the Pareto-based preference criterion of SPs as below.

Definition 1. *(Pareto-Based Preference Criterion) Suppose the current strategy of SP n, $a_n = \mathcal{F}_{m_1,k_1}$ and there exist a new strategy $a_n^* = \mathcal{F}_{m_2,k_2}$, then*

$$\begin{aligned} &\mathcal{F}_{m_2,k_2} \succ_n \mathcal{F}_{m_1,k_1} \Leftrightarrow u_n(a_n^*, a_{-n}) > u_n(a_n, a_{-n}) \\ &s.t. u_n(a_q, a_{-q}^*) \geq u_n(a_q, q_{-q}) \\ &\forall q \in \{\mathcal{F}_{m_1,k_1} \cup \mathcal{F}_{m_2,k_2}\} \backslash n \end{aligned} \tag{12}$$

This means that an SP will only prefer $\mathcal{F}_{m_2,k_2}$ over $\mathcal{F}_{m_1,k_1}$ if it can increase its profit after the switch and that the switch won't jeopardize the profit of any other SP in the two coalitions involved. Since only two coalitions are involved in a switch, the profit of the SPs in other coalitions won't be affected, making the switch a Pareto improvement of the system.

IV. Distributed coalition game-based algorithm

In this section, we will propose an efficient distributed algorithm to solve the coalition game and achieve a Nash-stable result.

At the start of the algorithm, the location of the servers and the users of the SPs are initialized. All the SPs originally have their services from the cloud. Each player will then have the chance to cache its service to an edge server by creating or joining a coalition. Members of each coalition can also change the number of VMs in their coalition if it will benefit each of the members. When no player has the incentive to deviate from its chosen strategy, the Nash equilibrium is found, and the algorithm will terminate. The detail of the algorithm is shown in Algorithm 1.

Theorem 1. *Algorithm 1 is guaranteed to converge to a final partition.*

Proof. According to the preference criterion defined in (12), a switch operation only occurs if it leads to a Pareto improvement, which means that each switch results in a new and unvisited partition with higher system utility. We know that the number of possible partitions of a set is a finite number known as the Bell number. Therefore, the algorithm is guaranteed to reach a final partition. □

Theorem 2. *The final partition of Algorithm 1 is Nash-stable.*

Proof. Assume the final partition of Algorithm 1 is not Nash-stable, which means that there exists at least one SP with the incentive to switch its coalition. However, after the SP switches, a new partition is created, which is contradictory to our assumption that the previous partition is the final partition. □

Algorithm 1 Distributed coalition game-based algorithm

```
Initialization:
    Set the positions of servers and SPs.
    Set the users for each SP
repeat
    Select an SP $n \in \mathcal{N}$ via a predetermined permutation
    and find its current coalition $\mathcal{F}_{m_1,k_1}$
    Uniformly randomly choose a server $m_2 \in M$
    Create an empty coalition in server $m_2$
    for all coalition $k_2$ in server $m_2$ do
        if $\mathcal{F}_{m_1,k_1} = \mathcal{F}_{m_2,k_2}$ then
            continue
        end if
        if $\mathcal{F}_{m_2,k_2} \succ_n \mathcal{F}_{m_1,k_1}$ then
            Add $\mathcal{F}_{m_2,k_2}$ to candidate list
        end if
    end for
    if candidate list is not empty then
        Make SP $n$ switch to the coalition in the candidate
        list that can make it reach the highest profit
    end if
    Delete empty coalitions
    Add or reduce the number of VMs in the coalition of
    SP $n$ if the profit of each member in the coalition will
    improve
until No SP has the incentive to change its strategy.
Output: The stable coalition formation result.
```

The profit maximization scheme is a mixed-integer nonlinear programming (MINLP) problem, which is NP-hard. The worst-case time complexity for each iteration in Algorithm 1 is $O(N^2)$, while the upper-bound of the number of iterations is exponential.

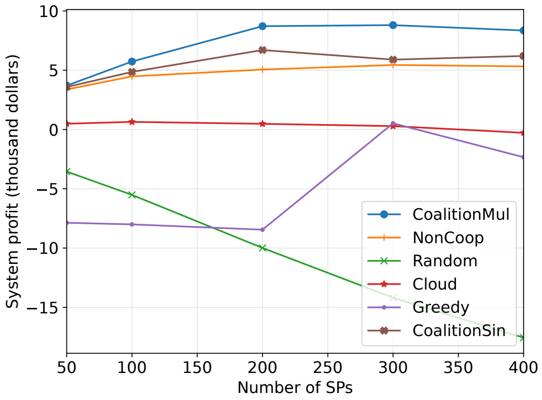

Fig. 2. System profit with different numbers of SPs.

V. Simulation Results

In this section, we analyzed the performance of the proposed scheme and compared it with other schemes through the numerical results of our simulation.

We consider 1 cloud server, M edge servers, and N SPs. Each SP has a random pool of users with 3 levels of delay requirements. The number of users for each level for each SP follows a uniform distribution $U(5, 20)$. The unit level price is \$2. The delay requirement for level 1 is $100ms$, and the delay requirement reduction for each level increase is $10ms$. The penalty to price ratio is 0.5. The request rate of each user is 0.5 requests per second, while the processing rate of each VM is 50 requests per second. The rent of a VM is \$70. The CPU unit limit of each edge server is 70, and the CPU unit allocated for each VM is 10.

Unless stated otherwise, the default parameter settings for N and M are provided as follow: $N = 100$, $M = 6$.

We compared the proposed scheme (`CoalitionMul`) with the following five schemes:

- Single coalition (`CoalitionSin`): Proposed in [10]. SPs using the same edge server can choose to collaborate or not, and those who choose to collaborate form a coalition. In other words, there will be at most one coalition for each edge server.
- Greedy delay (`Greedy`): Each SP greedily selects a coalition that will achieve the lowest delay and switches to it.
- Random (`Random`): Each SP randomly selects a coalition and switches to it.
- No cooperation (`NonCoop`): Each SP can only use its own leased edge server resources.
- No service caching (`Cloud`): All SPs have their services from the cloud server.

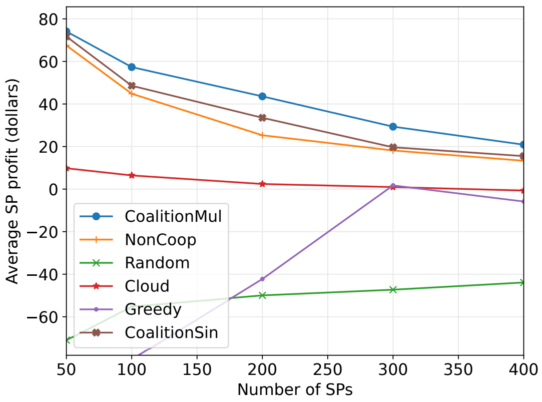

Fig. 3. Average SP profit.

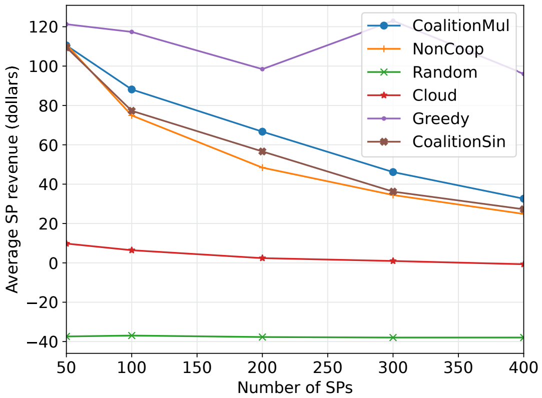

Fig. 4. Average SP revenue.

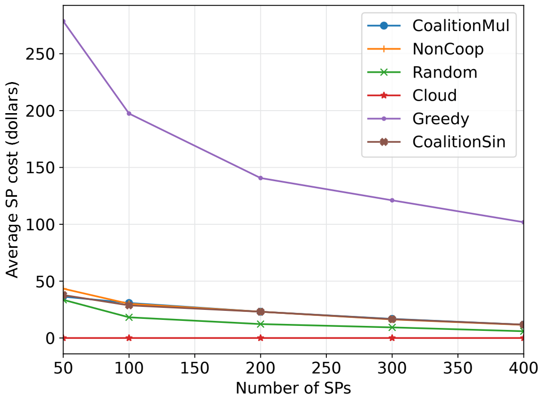

Fig. 5. Average SP cost.

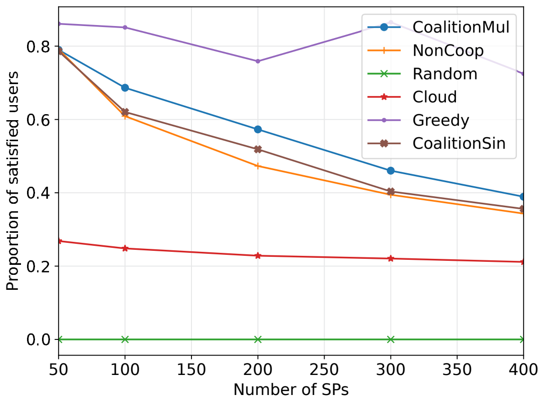

Fig. 6. Proportion of satisfied users with different numbers of SPs.

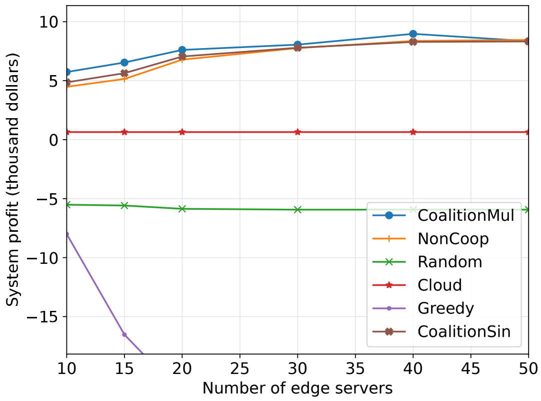

Fig. 7. System profit with different numbers of edge servers.

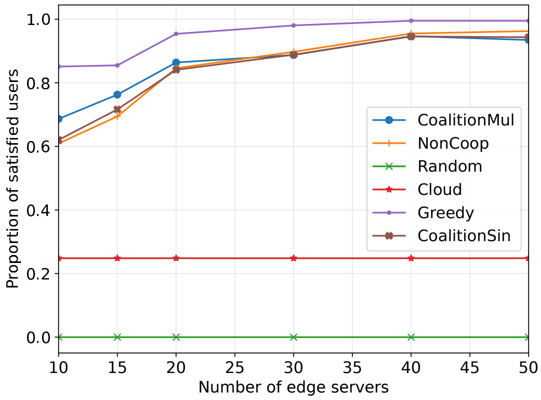

Fig. 8. Proportion of satisfied users with different numbers of edge servers.

A. Impact of profit from the number of SPs

Fig. 2 shows how the total system profit changes with different numbers of SPs. Although the system profit increases with the number of SPs in the proposed method, the average profit of each SP actually decreases when there are more SPs and users, as shown in Fig. 3. We know that profit is equal to revenue minus cost, so we can analyze it from these two aspects.

First, let's look at how revenue changes with different numbers of SPs as shown in Fig. 4. Revenue can be explained by the proportion of satisfied users as they're willing to pay their SPs once their delay requirements are satisfied, which is displayed in Fig. 6. When there are fewer service providers, an SP can easily find a small coalition on an edge server to switch to. But when the number of SPs increases, each coalition on edge servers becomes bigger, making the response time and the overall delay larger, resulting in more users receiving services that are over their delay requirement, leading to a decrease in the proportion of the users with their delay requirement satisfied.

Fig. 5 shows that the average cost decreases with a higher number of SPs. This can be explained by the increase in the average number of members in each coalition when there are more SPs, reducing the average cost for each member.

From the above analysis, it is shown that both the average revenue and the average costs drop with the increase in the number of SPs. However, the speed of the reduction of revenue is higher than that of the reduction of cost, so the average profit of an SP still decreases as the number of SPs increases.

From Fig. 6, we can see that `Greedy` outperforms our proposed algorithm in terms of the proportion of the users with their delay requirement satisfied. It is due to the fact that `Greedy` only values the delay of service, while the proposed algorithm also looks at the profit, including the revenue as well as the cost. As a result, even though `Greedy` can reach a higher level of user satisfaction, the profit achieved can never beat the proposed algorithm.

B. Impact of profit from the number of edge servers

Fig. 7 shows how the system profit changes with different numbers of edge servers. When the number of edge servers is still relatively small, the proposed algorithm reaches a higher system profit with more edge servers, as it is easier to reduce the delay when there are lots of choices of servers and coalitions for SPs to cache their services to. We can see this in Fig. 8. But at a certain point, the increase in system profit becomes minimal. The reason is that most of the users already have their delay requirement satisfied, so there is no incentive in further reducing the delay, as the SPs already get their money from the users.

Take a look at `Greedy` in both figures. The SPs in the greedy scenario always look for the lowest delay possible, regardless of the cost, even if their users are already satisfied. As a result, their profits keep dropping with the increase of choices, since they're willing to spend cash on the best choice possible for minimum delay.

C. Algorithms comparison

From Fig. 2 and Fig. 7, we can see that the proposed coalition scheme (`CoalitionMul`) always achieves the highest profit compared to the `NonCoop`, `Greedy`, `Random`, and `Cloud`, as it allows different SPs to share their VMs on the same edge server to make use of under-utilized resources and reduce costs. Since it allows SPs to form multiple coalitions in the same edge server, instead of limiting to one coalition, the proposed scheme also outperforms `CoalitionSin`.

As for the proportion of the users with their delay requirement satisfied, `Greedy`, which always looks for the lowest delay possible, outperforms the proposed algorithm, as the proposed algorithm also takes the cost of edge server resources into account and only views the proportion of satisfied users as a way to achieve higher revenue.

VI. Conclusion

In this paper, we investigated the collaboration between SPs through a game-theoretical mechanism. We model SPs as players trying to maximize their own profits by caching their service to an edge server to meet their users' delay requirement and form coalitions to share the cost of their VMs and make use of under-utilized resources. We proposed a coalition game-based algorithm to solve the problem. Simulation results showed that the proposed algorithm outperforms the other five algorithms, achieving 60% and 35% higher profit compared and to a non-cooperative scheme and a collaboration method proposed in a previous work respectively.

References

[1] D. Li, ”5G and intelligence medicine—How the next generation of wireless technology will reconstruct healthcare?”, Precis. Clin. Med., vol. 2, no. 4, pp. 205-208, Dec. 2019.

[2] ETSI, “Mobile edge computing (mec); framework and reference architecture,” ETSI, DGS MEC, vol. 3, 2016.

[3] S. -Y. Lien, S. -L. Shieh, Y. Huang, B. Su, Y. -L. Hsu and H. -Y. Wei, ”5G New Radio: Waveform, Frame Structure, Multiple Access, and Initial Access,” in IEEE Communications Magazine, vol. 55, no. 6, pp. 64-71, June 2017

[4] X. Wang, M. Chen, T. Taleb, A. Ksentini and V. C. M. Leung, ”Cache in the air: exploiting content caching and delivery techniques for 5G systems,” in IEEE Communications Magazine, vol. 52, no. 2, pp. 131-139, February 2014

[5] T. -Y. Kan, Y. Chiang and H. -Y. Wei, ”Task offloading and resource allocation in mobile-edge computing system,” 2018 27th Wireless and Optical Communication Conference (WOCC), 2018, pp. 1-4

[6] T. -Y. Kan, Y. Chiang and H. -Y. Wei, ”QoS-Aware Mobile Edge Computing System: Multi-Server Multi-User Scenario,” 2018 IEEE Globecom Workshops (GC Wkshps), 2018, pp. 1-6

[7] T. X. Tran and D. Pompili, ”Octopus: A Cooperative Hierarchical Caching Strategy for Cloud Radio Access Networks,” 2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 2016, pp. 154-162

[8] A. Mehrabi, M. Siekkinen and A. Ylä-Jaaski, ”QoE-Traffic Optimization Through Collaborative Edge Caching in Adaptive Mobile Video Streaming,” in IEEE Access, vol. 6, pp. 52261-52276, 2018

[9] Y. Chiang, C. -H. Hsu and H. -Y. Wei, ”Collaborative Social-Aware and QoE-Driven Video Caching and Adaptation in Edge Network,” in IEEE Transactions on Multimedia, vol. 23, pp. 4311-4325, 2021

[10] Z. Xu, L. Zhou, S. Chi-Kin Chau, W. Liang, Q. Xia and P. Zhou, ”Collaborate or separate? Distributed service caching in mobile edge clouds”, Proc. IEEE Conf. Comput. Commun. (INFOCOM), pp. 2066-2075, Jul. 2020.

[11] Ren, Ju and Liu, Jiani and Zhang, Yongmin and Li, Zhaohui and Lyu, Feng and Wang, Zhibo and Zhang, Yaoxue, ”An Efficient Two-Layer Task Offloading Scheme for MEC System with Multiple Services Providers,” IEEE INFOCOM 2022 - IEEE Conference on Computer Communications, 2022, pp. 1519-1528

[12] P1935. P1935 edge computing standard. [Online]. Available: https://standards.ieee.org/project/1935.html

[13] Shirin, A., and Puth, G. (2011). Customer satisfaction, brand trust and variety seeking as determinants of brand loyalty. African Journal of Business Management, 5.

[14] M. Mitzenmacher, E. Upfal. Probability and computing: randomized algorithms and probabilistic analysis. Cambridge University Press, 2005.

[15] W. Saad, Z. Han, M. Debbah, A. Hjorungnes and T. Basar, ”Coalitional game theory for communication networks”, IEEE Signal Process. Mag., vol. 26, no. 5, pp. 77-97, Sep. 2009.