IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received December 6, 2021, accepted January 11, 2022, date of publication January 19, 2022, date of current version January 28, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3144960

A Stackelberg Game-Based Dynamic Resource Allocation in Edge Federated 5G Network

JARGIS AHMED[1], MD. ABDUR RAZZAQUE[1], (Senior Member, IEEE), MD. MUSTAFIZUR RAHMAN[1], (Member, IEEE), SALMAN A. ALQAHTANI[2], (Member, IEEE), AND MOHAMMAD MEHEDI HASSAN[3], (Senior Member, IEEE)

[1]Green Networking Research Group, Department of Computer Science and Engineering, University of Dhaka, Dhaka 1000, Bangladesh
[2]Research Chair of New Emerging Technologies and 5G Networks and Beyond, Computer Engineering Department, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia
[3]Research Chair of New Emerging Technologies and 5G Networks and Beyond, Information Systems Department, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia

Corresponding author: Mohammad Mehedi Hassan (mmhassan@ksu.edu.sa)

This work was supported by the Deanship of Scientific Research at King Saud University through the Vice Deanship of Scientific Research Chairs: Research Chair of New Emerging Technologies and 5G Networks and Beyond.

ABSTRACT Mobile Edge Computing (MEC) plays key role for providing fast-response and high interactivity in the emerging 5G network. Edge service providers (ESP) are responsible for serving edge users or IoT devices running latency-critical applications. An MEC server provides faster response to ESPs but has limited computation resources, hence, it can be overloaded due to extensive resource demand. Thus, federation of multiple MEC servers offers an opportunity for dynamic resource allocation in a distributed manner. The federation objective is to maximize the usage of underutilized edge resources and reduction of service provision time simultaneously. As all the ESPs and MECs act autonomously, it is quite impossible for all the individuals to achieve optimal behavior simultaneously. In this paper, we develop a Stackelberg Game (*SBG*) based dynamic resource allocation method to reach the expected performance. The *SBG* analyzes the pricing of MECs and the resource-purchasing problem of ESPs. We also develop a many-to-many matching algorithm for resource sharing among the MECs and a one-to-many matching algorithm for that between an MEC server and ESPs. The results from an extensive performance evaluation demonstrate effectiveness of the proposed system in increasing utilities for MECs and ESPs, reducing the turnaround time of application tasks, and ensuring fair resource distribution compared to state-of-the-art works.

INDEX TERMS Edge federation, 5G network, mobile edge computing, resource allocation, Stackelberg game, utility maximization.

I. INTRODUCTION

The cloud computing and wireless communication integration created enormous scope for faster real-time communication. According to the current Cisco report [1], the annual global IP traffic will reach 396 exabytes (EB) per month which is 4.8 zettabyte (ZB) per year by 2022. Also, Traffic from wireless and mobile devices will account for 71 percent of total IP traffic by 2022. This traffic is the result of the 5G era. There are typically 3 scenarios present in 5G. The first one is enhanced Mobile Broadband (eMBB), which is characterized by high data rate requirements as virtual reality (VR), augmented reality (AR), etc. Second is Ultra-Reliable and Low Latency Communication (URLCC), characterized by latency intolerant services like self-driving cars [2]. The third one is the massive Internet of Things (mIoT) [3] which can be depicted as smart city, smart agriculture. Therefore, different applications require heterogeneous resources and it is quite challenging to meet the ongoing resource demand for mobile devices and IoT present at the 5G edge network [4]–[6]. Thus, the cooperation of Mobile Edge Computing (MEC) in terms of sharing resources is a challenging issue.

Enhancing the user's quality of experience (QoE) is very essential to ensure for diverse applications in 5G. Moreover, due to the increased computation requirement and limited computation and network resources, all tasks processing at local devices or offloading to the remote cloud for computation is inefficient and costly. In this case, the MEC will play

The associate editor coordinating the review of this manuscript and approving it for publication was Shadi Alawneh.

a pivotal role [7], [8] for communication and computation in 5G. The MECs provide not only computation capability and reduction of power consumption, but also shorten the computation and communication latency of IoT and mobile devices [9]. Most of the current research works investigate either to decrease the computation latency [9]–[11] for serving user tasks or computation resource optimization [12], [13]. However, the federation of MEC servers for resource sharing to serve edge users in the aid of resource shortage and to fulfill resource demand at the edge requires further exploration. Hence, the challenges for edge federation will be exploring resource availability among edge servers, the optimal requirement of resources, and the pricing for that. Also, dynamic resource allocation needs to be ensured in this scenario.

In the situation of resource shortage of any particular MEC server, the solution could be purchasing virtualized computational resource blocks (CRB) from different cloud servers [12]. However, in this case, the service latency may become intolerable, as ESPs are providing latency-critical applications, web services, etc. to edge users and IoT devices. Therefore, purchasing CRBs from other nearby MEC servers can ease the situation. Again, purchasing any amount of CRBs is not a practical approach to meet peak demand, as extra CRBs will remain unused. Hence, the MEC server must determine the optimal amount of CRB required by the ESPs as they are purchasing CRBs from MECs. Again, determining tolerable pricing of CRBs is required as ESPs will not buy CRBs at a higher price. When renting or federating resources, a buyer MEC server has to choose the best possible sellers for itself. In the same way, a particular seller can share resources with many buyers to increase its utility.

In this paper, we invade to overcome the above-mentioned issues. We consider edge federation for sharing abundant unused resources to aid overloaded MEC servers in a distributed manner. The target is to discover optimal resource optimization ensuring better pricing. This should maximize the usage of underutilized edge resources and reduce the service provision time simultaneously. The major contributions of this paper are listed as follows.

- We propose a distributed edge federation model for allocating resources dynamically between MEC servers and ESPs. The objective of each entity is to maximize its utility.
- We formulated the Stackelberg game to solve optimal requirement of resource and the pricing problem, observing the interaction between ESP and MEC servers. In the game, each MEC acts as a leader and sets the renting price for ESPs. The ESPs act as a follower and purchase the optimal amount of CRBs from MEC.
- After knowing the optimal demand of CRBs, MEC determines if it has to purchase resources from other MEC servers or not. Thus we propose a many-to-many matching for resource sharing among buyer MEC and seller MEC servers.
- MEC server serves multiple ESPs and an ESP can avail service from one MEC server. Hence, we also proposed a one-to-many matching between an MEC and ESPs to serve edge demand.
- Extensive simulation experiments were done on the *Cloudsim* [14] platform. The simulation results show the improvement in terms of the increased utility of MECs and ESPs, better service percentage of ESPs, and turnaround time of the tasks.

Rest of the paper is organized as follows. Section II is the literature review of the current scenario related to our work. Section III is the description of system model with assumptions and in Section IV, we formulated our problem. In Section V, we analyze the system based on the formulated problem and present our algorithms. Section VI elaborates the performance evaluation and finally, Section VII summarizes the paper.

II. RELATED WORKS

Cloud computing brought revolutionary in the context of computing and service management. The federation between clouds is the concept of cooperation between different cloud servers, which is a concept of service aggregation based on the interoperability features. There are two basic federation types [15], one is the vertical federation which spans multiple levels of the cloud stack, other is the horizontal federation which spans a single level of the stack. Therefore the authors in [16] considered horizontal federation focusing on the smaller and medium cloud servers to federate along with large ones to enhance the economical gain of the server capabilities. Also, there are issues like resources availability, robustness, price of the federated resource, and the revenue gained from federation along with maintaining the high QoS. In [17], authors have addressed these issues and provided a cloud federation model where they considered the horizontal federation of Infrastructure as a Service (IaaS) layer of different cloud servers.

As the current network is growing rapidly for connecting new devices entering every day for computation and communication, the static resource provisioning will not be sufficient enough to cope-up with the same rate. The software defined networking (SDN) and network function virtualization (NFV) enables flexible resource management and creates abstraction on the physical resource and internet services [18], [19]. Where NFV decouples network functions from dedicated hardware and SDN creates programmable centralized network control. The combination of SDN-NFV can be used as one of the key elements to make federation among clouds [20], [21]. In [22], the authors provided a centralized idea of SDN controller placed in MEC connected to the base station for the ultra-dense network. This SDN controller collects network status from a global perspective and advises mobile devices to perform tasks locally or offload tasks to a nearby small cell base station. An approach of distributed computation offloading and profit maximization is done in [13]. Where SDN controller placed in MEC and it controls only the resources required for different mobile virtual network operators (MVNOs) and offloading process from user to MEC with an extra element called virtualized user (VU). The joint optimization of the model considering the VU, MEC, and MVNOs are proposed by the authors and their proposed algorithm reduces the signaling overhead and complexity of the computation.

In [23], authors propose an auction based model for resource sharing between edge servers and the service providers to maximize the utility of each entity. Here, the allocation of resources is distributed and this process is also decoupled from service provisioning management at the service provider end. Now, when one particular MEC has abundant resources, it may share them with the cloud server and other edge servers also. On the other hand, when the demand increases that particular MEC have to buy resources from cloud server as well as from MEC servers. In [12], authors presented an effective process of wholesaling abundant edge resources to cloud server and buyback again if required by a particular MEC. Here, the main objective was to increase profit for MEC by wholesaling its resource which is underutilized at a lower price, and then buyback at a higher price if required to serve incoming tasks of that MEC server. In [24], the authors discussed the issue of sharing the resources between different MEC servers. They proposed a cooperative game based framework, where the edge servers form a coalition for resource sharing. When one MEC can not meet the resource requirement for its native applications other MECs can aid this situation by providing resources after ensuring the fulfillment of their requirements. They provided Pareto optimal solution for allocating resources using the cooperation of the MEC servers present in the system.

Most of the existing studies do not focus on the impact and opportunity of the federation of edge servers to handle extensive edge resource demand. Few literature works focused on the sharing resources between edge to cloud servers [12] and edge to edge servers [24], although none of the works focus on the optimal resource requirement of the ESPs to serve edge users and the objective of maximizing the utility. When the ESPs resource demand information at MEC servers are properly available, then it eases the decision for MEC to buy or sell resources. Hence, this observation lead us to design a Stackelberg game based approach for finding the optimal behavior of ESPs and MEC servers and federate resources between preferred MEC servers.

III. SYSTEM MODEL AND ASSUMPTIONS

In the edge network, the edge user runs different types of applications where most of which are latency-critical and requires real-time interactions. The Edge Service Provider(ESP) consists of User Equipment or IoT devices for smart-home. To serve the incoming tasks, ESP requests resources from one particular InP. The InP consists of a base station, MEC server, and SDN controller. For the case of simplicity, we use characterize InP as a MEC server. One MEC

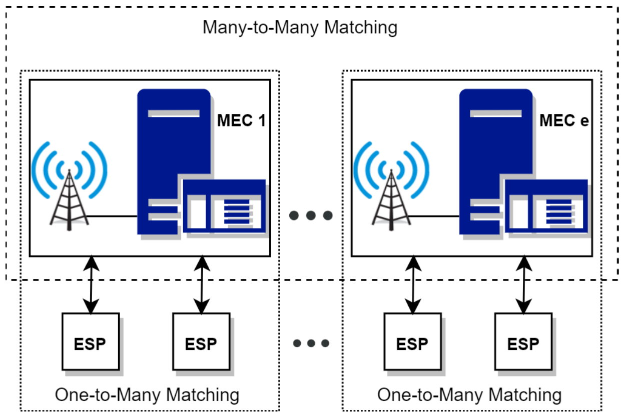

FIGURE 1. 5G edge network federation architecture.

server is connected with other MEC servers. The figure (1) shows the federated environment of edge federation.

Let the set of ESP is $\mathcal{M} = \{1, 2, \ldots, m\}$ and the number of ESP is M. The set of InPs containing the MEC server is $E = \{1, 2, \ldots, e\}$. For serving edge users or IoT devices, each ESP requests resources from its nearby MEC server. We define the unit amount of resource that can be purchased by ESP from the MEC server as a computational resource block (CRB). The service rate of each CRB is denoted as μ. A particular MEC can serve multiple ESP by its resources. The aggregated tasks arrival rate to m^{th} ESP is represented by λ_m. Now, if a large number of ESPs prefer to get CRBs from one particular MEC server to serve its incoming tasks, that MEC will face resource scarcity. Therefore, this MEC can now act as a buyer and buy CRBs by the federation process, from other MEC servers who have abundant CRBs to sell. S^e is the set of sellers maintained by buyer MEC e, where the total number of sellers is N. A seller n in practicality cant serve all its buyers. Therefore, it will maintain a preference list of the buyer MECs, denoted as $\mathcal{L}^n$ from its buyer set $\mathcal{B}^n$ containing B number of buyers. Also, a buyer MEC can buy resources from multiple sellers. Hence, it also maintains a preference list of sellers $\mathcal{L}^b$. Furthermore, e^{th} MEC also has its preference to serve connected ESP, where the list is denoted by $\mathcal{L}^e_m$

As already mentioned ESP serves UE and IoT devices running real-time applications, better QoS must be ensured for edge service. The measurement of QoS for m^{th} ESP depends on the service delay t_m which is calculated by

$$t_m = f_m + d_m, \tag{1}$$

where f_m is the network delay and d_m is the queuing delay. According to [25] the queuing delay for serving ESP m can be calculated by the following equation

$$d_m = \frac{\lambda_m}{\mu - \frac{\lambda_m}{q_m}}, \tag{2}$$

where q_m is the amount of CRBs purchased by the ESP m. The network delay can be influenced by the traffic situation of the network, distance for transmitting, and many other uncertain

TABLE 1. List of notations and definitions.

Notation	Definition
$\mathcal{M}, E$	The set of ESPs and MEC servers, respectively
λ, μ	Task arrival and service rates, respectively
$S, \mathcal{B}$	The set of seller and buyer MECs, respectively
$\mathcal{T}$	Tolerable delay deadline
t, f, d	Service, network and queuing delays of an ESP
q_m	Amount of CRBs purchased by the ESP m
r_e	Per unit price of virtualized CRBs
l_m^e	Distance from ESP m to the MEC e
$\mathcal{U}_m$	Utility function of ESP m
$\mathcal{U}^e$	Utility function of buyer MEC e
$\mathcal{U}^n$	Utility function of seller MEC n
α, β, γ	Weight factors used by each ESP utility function
ω_m^e	The association indicator of ESP
q_n^e	Amount of CRBs purchased form seller n
p_n^e	Price set by seller MEC n
t_n^e	Communication delay between buyer e and seller n
q_e^r	Amount of CRBs are reserved by MEC e
ε_e	The energy cost of serving each CRBs MEC e
p_n^i	Price set by the seller n for buyer i
q_n^i	Amount of CRBs served to a buyer i from seller n.
$\mathcal{L}_N^e$, $\mathcal{L}_i^n$, $\mathcal{L}_M^e$	Preference list of a buyer MEC e, seller MEC n and ESPs respectively

factors. However, for the case of simplicity, the dependency of latency for network delay is greatly dependent on the total distance l_m^e from ESP m to the physical service provider e. Therefore f_m has a linear relationship with l_m^e and scalar θ represented as

$$f_m = \theta l_m^e. \tag{3}$$

To serve the edge users, ESP have to pay for purchasing CRBs from MEC, following the model presented in [26], the utility of ESP can be measured by the revenue received from the incoming workload of ESP and subtracting the cost for the payment of MEC and the cost incurred by the service delay, represented as following

$$\mathcal{U}^m = \alpha_m \lambda_m - \omega_m^e \beta_m q_m r_e - \gamma_m t_m, \tag{4}$$

where α_m, β_m, γ_m are the weight factors determined by each ESP for itself. ω_m^e is the association indicator where $\omega_m^e = 1$ indicates ESP m is connected to MEC e and 0 otherwise. Also one ESP can avail service form one MEC only. r_e is the per unit price of virtualized CRBs set by MEC e and q_m is the amount of CRBs purchased by m form e.

According to the purchasing request from the serving ESP, each MEC calculates its utility from revenue gained from serving ESPs minus the energy cost for serving CRBs by itself. However, if e does not have a sufficient amount of CRBs present at the moment, it needs to buy CRBs from other seller MECs. Therefore the payment for buying CRBs and cost for communication delay from different sellers also subtracted from the utility of e, which is

$$\mathcal{U}^e = \sum_{m=1}^{M} \omega_m^e q_m r_e - \sum_{n=1}^{N} (p_n^e q_n^e + \psi^e t_n^e) - \varepsilon_e q_e^r, \tag{5}$$

where q_n^e is the amount of CRBs purchased from seller n, p_n^e is the price set by n, ψ^e is the weight factor and t_n^e is the communication delay between e and n. q_e^r is the number of CRBs is reserved by that particular MEC e and ε_e is the energy cost of serving each CRBs by itself.

Again from the perspective of a seller MEC the utility function will be the payment received from the buyers minus the maintenance cost for serving each buyer as follow

$$\mathcal{U}^n = \sum_{i=1}^{B} q_n^i (p_n^i - c_n), \tag{6}$$

where p_n^i is the price set by the seller n itself, c_n is the maintenance cost and q_n^i is the amount of CRBs rented by a buyer i from n.

IV. PROBLEM FORMULATION

According to the federation architecture between MECs, the interactions of resource trading between ESP and MEC and between buyer MEC to seller servers, all the individual ESP, buyer MEC, and seller servers can't achieve the maximal utility simultaneously. Therefore we need to observe a sequential decision making process. In this process, MEC predicts the total number of CRBs purchased by its serving ESPs and solves the pricing problem for ESPs using Stackelberg Game. Observing the behavior of MEC, each ESP determines to buy resources from MEC. After that, a buyer MEC now knows the amount of CRBs it needs to buy from seller servers and requests accordingly. Finally, each seller will provide the required amount of CRBs to its buyer and compete with other seller servers.

The objective of ESP is to maximize its utility by serving the edge user and IoT devices. Therefore, we can write the optimization problem for each ESP as follows

$$\underset{q_m}{\text{maximize}} \ \mathcal{U}^m(q_m|r) \tag{7}$$

$$\text{subject to:} \ \sum_{e=1}^{E} \omega_m^e = 1 \tag{8}$$

$$t_m \le \mathcal{T} \tag{9}$$

$$\alpha_m \lambda_m \ge \beta_m q_m r_e + \gamma_m t_m \tag{10}$$

$$q_m \ge 0 \tag{11}$$

where the pricing profile r is the price value set by all the MEC servers nearby to the ESP m. Constraint (8) ensures that ESP can avail service from only one MEC. Constraint (9) indicates tasks service delay should not exceed delay deadline. Constraint (10) indicates utility gained from edge users should be greater than the resource buying utility and constraint (11) ensures that ESPs received nonnegative resources from the MEC server.

After observing the behavior of ESPs, MEC now sets the serving price to serve ESPs and compete with other MECs. Also, all MEC will determine if it needs to buy CRBs from seller servers or not. Therefore the optimization problem for the buyer MEC is

$$\underset{r_e}{\text{maximize}} \ \mathcal{U}^e(r_e|Q_e^*, P, r_{-e}^*) \tag{12}$$

$$\text{subject to:} \quad \sum_{m=1}^{M} \omega_m^e q_m r_e \geq \sum_{n=1}^{N} (p_n^e q_n^e + \psi^e t_n^e) + \varepsilon_e q_e^r \tag{13}$$

$$t_m + t_n^e \leq \mathcal{T} \tag{14}$$

$$r_e \geq 0 \tag{15}$$

$$q_n^e \geq 0 \tag{16}$$

$$q_e^r \geq 0 \tag{17}$$

where Q_e^* denotes the optimal amount CRBs purchased by all the connected ESPs. P is the rent profile of all the seller servers. r_{-e}^* is the optimal price profile, set by other MECs except e. The constraint (13) ensures that utility gained by serving ESPs should be greater or equal to the resource buying utility and energy utility. Service latency should not exceed the delay deadline of tasks, hence, the buyer will buy resources from the seller where the constraint (14) can be met. Constraints (15, 16, 17) indicates non negative price and non negative resource purchased from sellers and MEC allocates non negative resources to ESPs respectively.

A seller MEC also wants to maximize its utility, therefore, it chooses the most preferable buyer MECs. Also, it needs to compete to sell its CRBs to buyers. The optimization problem for the seller server is

$$\underset{q_n^i}{\text{maximize}} \;\; \mathcal{U}^n(q_n^i | Q_{-i}^*, P_{-n}^i) \tag{18}$$

$$\text{subject to:} \;\; p_n^i \geq c_n \tag{19}$$

$$q_n^i \leq q^{i-th} \tag{20}$$

$$\sum_{i=1}^{B} q_n^i \leq Q_n^s \tag{21}$$

where Q_{-i} is the optimal amount of CRBs required by buyers other than i. P_{-n}^i is the pricing profile set by other sellers except n. The constraint (19) indicates pricing of CRBs should be greater or equal to the maintenance cost. Constraint (20) ensures allocation of resource to buyer i will be smaller or equal to the demand q^{i-th}. Also, a seller will allocate CRBs to buyer MECs up to the available amount Q_n^s, which is represented by constraint (21).

After observing the interactions between ESPs, MECs and seller servers, we can focus on the following problems

1) **Amount of resource purchased by ESP from MEC:** ESPs are serving latency-critical applications of edge users and IoT devices. Again, the demand for CRBs is influenced by the service delay and pricing of the MEC server. Hence, the number of CRBs required by the ESPs should be optimal to increase utility.
2) **Determining the price for ESP by MEC:** When MEC provides CRBs to ESPs, the pricing of the CRBs should not be intolerable. Because at higher price ESPs will buy less resource from that MEC server and will give up if it can not gain positive utility. Also, MEC has to bear the serving cost and resource buying cost if it acts as a buyer. Hence, MEC has to ensure the optimal pricing of CRBs.
3) **Resource allocation problem between buyer-seller:** When a MEC needs to buy resources from seller servers, it does have some preferences for selecting sellers. In the same way, each seller wants to allocate CRBs to its buyers to increase utility value. To maximize the utility, each seller will also have a preference list for selecting its buyers.
4) **Resource allocation problem for MEC-ESP:** MEC will allocate resources to its serving ESPs after matching between MEC and all its ESPs. MEC will prefer the ESPs form which the total utility will be maximized based on the pricing to ESP.

According to the above formulated problem and discussion, we observe that each ESPs, MECs are autonomous in their actions. Each individual observes other actions and takes the optimal decisions to improve their utility. Therefore to reach the optimal behavior, we first formulate the Stackelberg game where the MEC server acts as a leader and ESPs acts as followers. The purpose of the game is to solve the pricing problem for MECs and the resource purchasing problem for the ESPs. After calculating the optimal number of CRBs to be purchased by ESPs, the MEC decides to buy external resources if required, and runs a many-to-many matching algorithm. Finally, to serve ESPs by a particular MEC server a one-to-many matching algorithm is proposed. Based on the game, to determine the optimal resource demand and pricing, we consider the following lemma.

Lemma 4.1: The Stackelberg game between MEC and the ESP, when MEC sets the service price r_e, the optimal amount of CRBs required to maximize utility of a ESP is

$$q_m^* = \frac{\lambda_m}{\mu}\left(1 + \sqrt{\frac{\gamma_m}{r_e \beta_m}}\right) \tag{22}$$

Proof: Based on the utility equation of ESPs in eq. (4) the first derivative of $\mathcal{U}^m$ with respect to q_m is

$$\frac{\partial \mathcal{U}^m}{\partial q_m} = -r_e \beta_m + \gamma_m \left(\frac{\lambda_m}{\mu q_m - \lambda_m}\right)^2 \tag{23}$$

Also the second derivative of $\mathcal{U}^m$ with respect to q_m is as follows

$$\frac{\partial^2 \mathcal{U}^m}{\partial^2 q_m} = -\frac{2\gamma_m \mu \lambda_m^2}{(\mu q_m - \lambda_m)^3} \tag{24}$$

Now as the $\frac{\partial^2 \mathcal{U}^m}{\partial^2 q_m} < 0$, which indicates that $\mathcal{U}_m^e$ follows a decreasing relationship with respect to q_m. Hence if we consider the first derivative $\frac{\partial \mathcal{U}^m}{\partial q_m} = 0$, we will get the optimal number of CRBs or q_m^* purchased by each ESP m. □

As the actions of ESPs regarding the optimal CRBs are known to MEC server, the adjusted optimization function for a particular MEC server e is as follows

$$\underset{r_e}{\text{maximize}} \quad \widehat{\mathcal{U}}^e(r_e | Q_e^*, P, r_{-e}^*) \tag{25}$$

$$\text{subject to:} \tag{26}$$

$$\sum_{m=1}^{M} \frac{\lambda_m}{\mu}\left(1 + \sqrt{\frac{\gamma_m}{r_e \beta_m}}\right)\omega_m^e r_e \geq \sum_{n=1}^{N} (p_n^e q_n^e + \psi^e t_n^e) + \varepsilon_e q_e^r \tag{27}$$

and constraints 14-17, where the value of $\mathcal{U}^e$ is updated to $\widehat{\mathcal{U}}^e$ as follows,

$$\widehat{\mathcal{U}}^e=\sum_{m=1}^{M}\frac{\lambda_m}{\mu}\left(1+\sqrt{\frac{\gamma_m}{r_e\beta_m}}\right)\omega_m^e r_e-\sum_{n=1}^{N}(p_n^e q_n^e+\psi^e t_n^e)-\varepsilon_e q_e^r. \tag{28}$$

We observed that the first derivative of $\widehat{\mathcal{U}}^e$ with respect to r_e is greater than 0, where

$$\frac{\partial\widehat{\mathcal{U}}^e}{\partial r_e}=\frac{\lambda_m}{\mu}\left(1+\frac{1}{2}\sqrt{\frac{\gamma_m}{\beta_m r_e}}\right), \tag{29}$$

which indicates that the utility of a MEC server e is positive co-related with the pricing set by e to its ESPs. Again the as the service delay for any particular ESPs can not cross $\mathcal{T}$, therefore the low bound for purchasing CRBs by any ESPs given as follows

$$q_m\geq\frac{\lambda_m\mathcal{T}}{\mu\mathcal{T}-\lambda_m}. \tag{30}$$

Finally following the low bound and eq. (22), the optimal pricing by MEC e to its ESPs for maximizing its utility is

$$r_e=\frac{\gamma_m}{\beta_m}\left(\frac{\mu\mathcal{T}-\lambda_m}{\lambda_m}\right)^2. \tag{31}$$

After calculating the optimal resource demand at the edge and pricing for ESPs, now we have to allocate resource dynamically.

V. DISTRIBUTED ALGORITHM FOR RESOURCE FEDERATION AT THE EDGE

After observing the behavior of ESPs, one particular MEC knows the amount of required CRBs (Q_e) it needs to purchase from nearby seller servers. To find out the best possible seller, each buyer needs to make their preference list. Each buyer MEC wants to buy resources from the seller considering lower renting price and minimal communication latency. Therefore the preference list of a particular MEC e who needs to buy CRBs form sellers is $\mathcal{L}_N^e=\{\mathcal{L}_1^e,\mathcal{L}_2^e,\ldots\mathcal{L}_n^e\}$ considering,

$$\mathcal{L}_n^e=-(\zeta^e\mathcal{P}_n^e+\eta^e\tau_n^e), \tag{32}$$

where ζ^e and η^e are weight factors set by each buyer for itself, $\mathcal{P}_n^e$ and τ_n^e are the normalized price and communication delay, respectively.

Accordingly, a seller server n will make its preference list of buyers $\mathcal{L}_I^n=\{\mathcal{L}_1^n,\mathcal{L}_2^n,\ldots\mathcal{L}_i^n\}$. Each seller prefers the buyers from which it can gain more profit thus,

$$\mathcal{L}_i^n=p_i^n-c^n. \tag{33}$$

After the construction of the preference list by each buyer, it will try to buy resources from its preferred sellers. In Algorithm 1 we set a pointer for each buyer MEC to point its preferred seller from the preference list. At the starting phase, the pointer will point to the first seller. In each iteration, the buyer will propose its preferred seller for sharing resources and wait for confirmation. We also set a flag f_e for each buyer MEC, which indicates wheater the seller n allocates its resource to that buyer or not. Initially, the value is set to 1 and when a buyer proposes a seller n, the value of f_e will be 0. Now, if one particular buyer is rejected by seller n, it will update its pointer to point next most preferred seller form $\mathcal{L}_N^e$ and propose in the same way. This process of requesting seller continued until the demand of total amount of required CRBs Q_e met.

Algorithm 1 Many-to-Many Matching Resource Sharing for Buyer MEC

```
    Input: Seller MEC information
    Output: Matched sellers
 1: Buyer $e$ constructs preference list of sellers $\mathcal{L}_N^e$ according
    to Eq. (32)
 2: To act as an indicator one pointer is set to point at the first
    seller in $\mathcal{L}_N^e$
 3: Set flag $f_e$ to indicate whether $e$ has been chosen by $n$,
    initially $f_e = 1$
 4: while all seller from $\mathcal{L}_N^e$ not scanned by $e$ do
 5:     if $f_e = 1$ then
 6:         Pointer stays current position in $\mathcal{L}_N^e$
 7:     else
 8:         Pointer iterates next preferred seller in $\mathcal{L}_N^e$
 9:     end if
10:     propose to pointed seller $n$ according to preference for
        purchasing CRBs
11:     $f_e \longleftarrow 0$
12: end while
```

From the perspective of a seller in Algorithm 2, based on the proposals from buyers, each seller constructs buyer preference list $\mathcal{L}_I^n$, and allocate its resource for rent to the most preferred buyer. After allocating CRBs to one buyer, the pointer of the seller updates to point next most preferable buyer in the next iteration. Selection of buyer will happen until the total available amount of CRBs Q_n^s of the seller n is not 0. If there are no more available CRBs to share by the seller to its buyer, it simply rejects the requests.

On the other hand the MEC will choose its serving ESPs based on the preferences. A particular MEC server will construct its own preference list as $\mathcal{L}_M^e=\{\mathcal{L}_1^e,\mathcal{L}_2^e,\ldots,\mathcal{L}_m^e\}$ where,

$$\mathcal{L}_m^e=r_e. \tag{34}$$

After the federation process of renting resources from different sellers, each MEC e will allocate its currently available number of CRBs Q_e to its serving ESPs.

The allocation of CRBs by MEC server to ESPs depicted in Algorithm 3. MEC will construct the preference list of ESPs and set a pointer for each MEC to point ESPs in $\mathcal{L}_m^e$. Initially, the pointer points first ESP. We also set flag f_m to indicate if ESP m is chosen or not. Initially, the pointer value is 1. Each MEC selects its serving ESPs based on the preference

Algorithm 2 Many-to-Many Matching Resource Sharing for Seller MEC

```
Input: Buyer MEC requests
Output: Matched buyers
1: Seller $n$ constructs preference list of buyers $\mathcal{L}^n_I$ according to Eq. (33)
2: while $\mathcal{Q}^s_n \neq 0$ do
3:     if buyer $i \in S_i$ selected according to the $\mathcal{L}^n_I$ then
4:         if $\mathcal{Q}^s_n > q_i$ then
5:             Allocate $q_i$ to $i^{th}$ buyer
6:             $\mathcal{Q}^s_n \longleftarrow \mathcal{Q}^s_n - q_i$
7:         else
8:             Allocate $\mathcal{Q}^s_n$ to $i^{th}$ buyer
9:             $\mathcal{Q}^s_n \longleftarrow 0$
10:        end if
11:    else
12:        Reject buyer $i$ and set flag $f_i \longleftarrow 1$
13:    end if
14:    Pointer moves to next preferred buyer
15: end while
```

Algorithm 3 One-to-Many Matching Algorithm for an MEC

```
Input: Requests from ESPs
Output: Matched ESPs
1: MEC $e$ constructs $\mathcal{L}^e_M$ for all ESPs following Eq. (34)
2: To act as an indicator one pointer is set at the first ESP in $\mathcal{L}^e_M$
3: Set flag $f_m$, $\forall m \in \mathcal{M}$ to indicate whether $m$ has been chosen by $e$, initially $f_m = 1$
4: while all $m$ is not scanned from $\mathcal{L}^e_M$ do
5:     if $Q_e \geq q_m$ then
6:         Allocate $q_m$ to ESP $m$
7:         $Q_e \longleftarrow Q_e - q_m$
8:         $f_m \longleftarrow 0$
9:     else
10:        Allocate $Q_e$ to ESP $m$
11:        $Q_e \longleftarrow 0$
12:        $f_m \longleftarrow 0$
13:    end if
14:    Pointer moves to next preferred ESP where $f_m = 1$
15: end while
```

and allocate resource according to the requirement and set $f_m = 0$. If one particular MEC can not fulfill all ESPs resource demands then it prefers to serve only the most preferable ESPs and low preferred ESP will not get the desired amount of CRBs.

VI. RESULT ANALYSIS

In this section, we discuss the simulation setup and the analysis of the proposed edge federation process in detail. We used *CloudSim* [14] for the implementation and simulation result analysis. For simplification, we denoted our proposed method as Stackelberg Game (SBG). We compare our SBG with standalone non-federated (NF) MECs and another state-of-the-art-work Cooperative Game(CG) [24].

A. SIMULATION ENVIRONMENT

We considered there are 3-9 MEC servers, and 200 - 320 ESPs randomly in a circular distance of 100km. By following the practical environmental settings in [25], we set the tasks arrival rate of each ESP λ_m as a random number having the average of 0.5 per millisecond (ms) which is 500 per second (s). The CRB service rate is 0.1 $(ms)^{-1}$ or 100 per second. The maximum delay deadline for all ESPs is 60 ms. Based on the usage of its computing resources and its service cost, we set the announced rent as a random number satisfying the uniform distribution between (5, 10) also the announced price for buyer MEC is between (0, 5), and the amount of available CRB as a random number satisfying uniform distribution between (200, 400). The weight factors α_m, β_m, γ_m are set as 20, 0.01 and 0.001, respectively. Performance parameters can have values as summarized in Table 2.

TABLE 2. Simulation parameters.

Parameters	Value
Number of ESPs	$200 \sim 320$
Number of MECs	$3 \sim 9$
Tasks arrival rate at ESP (λ_m)	$200 \sim 800$ per second
Service rate of CRB (μ)	100 tasks per second
Available CRB at MEC	$200 \sim 400$
Size of CRBs, tasks size	1000 MIPS, 2000 MIPS
Maximum delay deadline ($\mathcal{T}$)	60 ms
$\alpha_m, \beta_m, \gamma_m$	20, 0.01, 0,001
Distance of ESP from MECs	$0 \sim 100$ km
Announced price of MEC for ESP	$5 \sim 10$
Announced price for buyer MEC	$0 \sim 5$

B. PERFORMANCE METRICS

1) *Average utility of MECs* is defined as the average utility achieved by MECs, providing the virtualized CRBs for serving ESPs
2) *Average utility of ESPs* is the average utility gained by ESPs as it serves edge users or IoT devices.
3) *Percentage of served ESPs* is the measurement of the ESPs, which receives the required amount of CRBs from the requesting MEC.
4) *Average turnaround time of tasks* is calculated for successful completion of tasks, which arrived at each ESPs.

C. SIMULATION RESULTS

We study the performance of our proposed *SBG* method by varying the number of ESPs, changing the task arrival rate at ESPs, and varying the number of MECs.

1) IMPACT OF VARYING THE NUMBER OF ESPS

The Fig. 2 depicts the impact of varying the number of ESPs in the system. To study the impact we change the number of ESPs from 200 to 320 and assume the tasks arrival rate at each

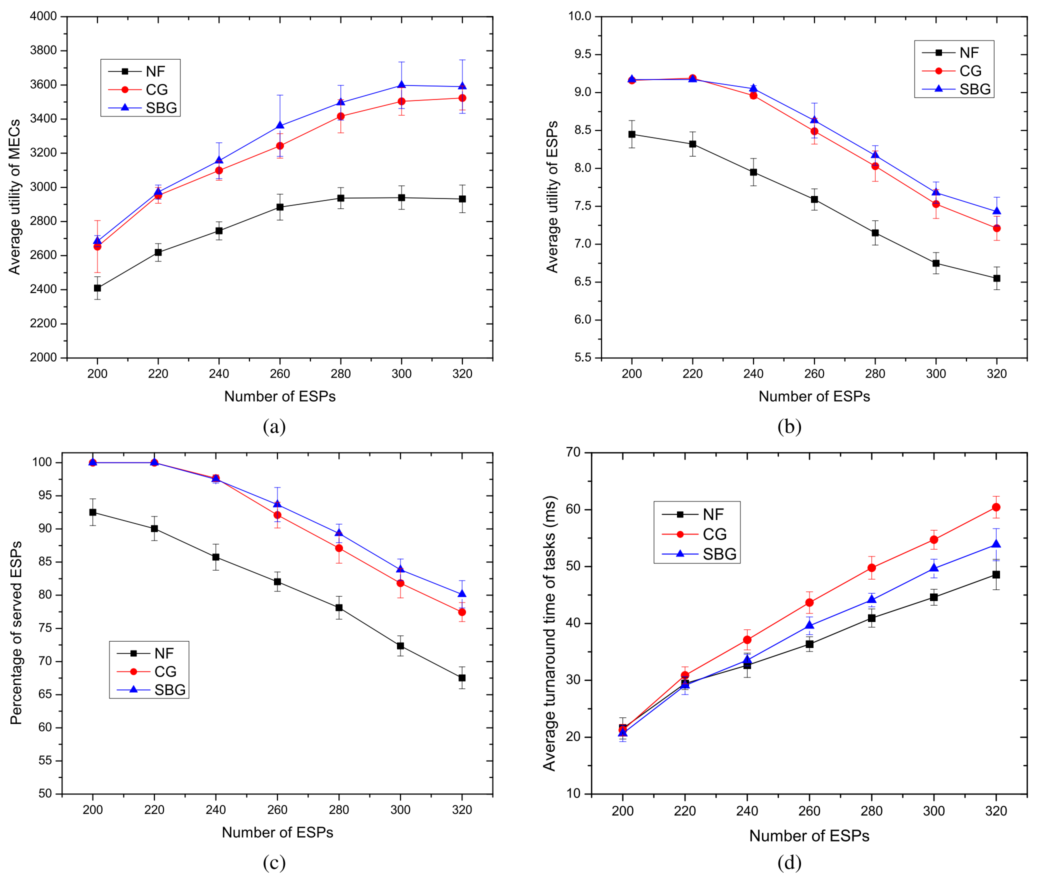

FIGURE 2. Impact of varying the number of ESPs on the performance of the studied systems.

ESP is 500 per second and 5 MECs for providing resources to the ESPs.

In Fig. 2a, we can observe that as the number of ESP increases the average utility of MECs increases. The utility increases rapidly for MECs when the number of ESPs is 200 to 280, after that the growth of utility gradually decreased to the nearly equal utility for the last few points. However, the saturation point is reached when the number of ESP is 300 to 320. The reason behind this is when all MEC have available resources, increasing ESP helps MECs to gain more utility by providing its resources to the demanding ESPs according to their requirements. After when the ESPs number reached 300 to 320, all MECs are now saturated by demanding ESPs and MECs have no more resources available to provide to ESPs. We can also observe that our proposed *SBG* and the method *CG* have a very close average utility of MECs compared to the *NF* method. The reason behind this is in *SBG* a buyer MEC and buy CRBs from other sellers and in *CG* all the MECs which have abundant resources can share resources ESPs other than its serving ESPs. On the contrary in the *NF* method, an overloaded standalone MEC can't explore abundant resources of other MECs. Therefore MECs in *NF* can not gain utility by sharing or directly selling CRBs to ESPs. However, our proposed SBG outperforms both the *CG* and *NF* methods.

In Fig. 2b, the average utility of ESPs is higher when the number of ESPs was low and decreased drastically according to the increasing number of ESPs. The reason behind this is when the number of ESPs increases MECs are providing available CRBs to ESPs and can satisfy the total amount of resource demand up to a certain point. After that increasing ESPs actually increases the demand of CRBs and now it will cause resource scarcity, therefore the number of fully served ESPs will decrease. In this scenario, each ESPs have to wait more and the competition of getting resource is higher for all the described methods. As usual, the *NF* method performs worst and our proposed *SBG* outperforms the other two methods of resource provisioning.

As already mentioned that, an increasing number of ESPs will cause resource starvation and eventually decrease the average utility of ESPs; the percentage of served ESPs will also decrease. In Fig. 2c, we observe that when the ESPs number in the system is low link 200 to 220 both *SBG* and *CG* can serve 100% of ESPs according to the demand. When ESPs increases the percentage of served ESPs decreases accordingly. The *NF* performs worst as in this approach, MECs can not buy or federate resources from other MECs hence the percentage of ESPs getting served is very low compared to the other methods and *SBG* outperform *CG*.

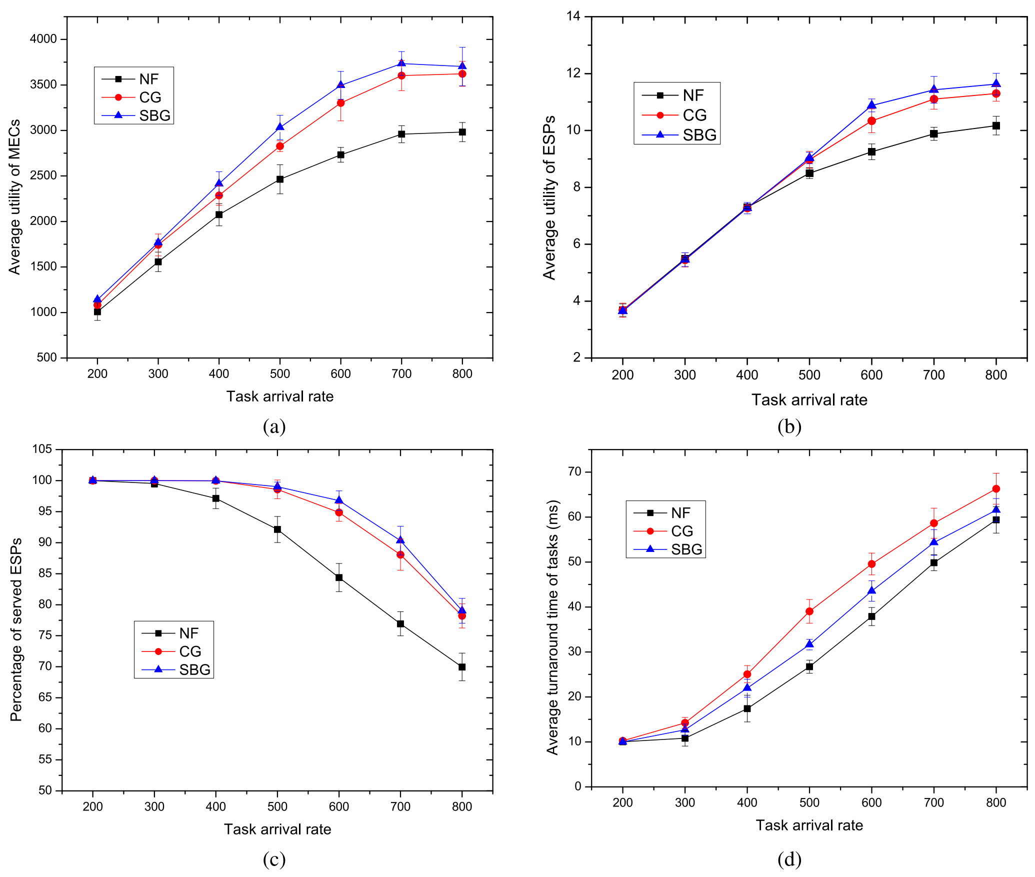

FIGURE 3. **Impact of varying task arrival rate on the performance of the studied systems.**

In Fig. 2d, we observe the average turnaround time (TAT) of successfully completed tasks. We can find that the increasing number of ESPs increases the TAT of tasks in all the described methods. We observe that the *NF* method has the lowest TAT whereas the *CG* method has the highest TAT value. The reason behind this is, in the *NF* method the MECs do not communicate with other MECs to share resources that's why the burst time is less than other methods. Whereas in *CG* ESPs have to wait until another MEC who have abundant resources decides to share its CRBs to that particular ESP which are not getting resources. In contrast, in the *SBG* method if a MEC faces resource shortage, that particular MEC by itself request other MECs to share resources, and thus *SBG* requires less time for finding resources and satisfying it's demanding ESPs requests. We also can observe that when the number of ESPs much higher like 300 or 320 the TAT is much better in our proposed *SBG* method. Although it seems *NF* is better than the other two approaches, but we can surely say that is not true when we observe in Fig. 2c.

2) IMPACT OF VARYING TASK ARRIVAL RATES

In Fig. 3 we measure the impacts of changing tasks arrival rate at each ESPs. We change the arrival rate from 200 to 800 tasks per second and maintain the number of ESPs present in the system is 200 also the number of MECs are 5, and observe the impacts.

In Fig. 3a, depicts that when the average tasks arrival rate of each ESP increases the average utility of MECs increases. The reason behind this is now MECs can increase prices for CRBs and ESPs have to buy them at a higher price. The utility value is increasing up to tasks arrival rate 700 for all the *NF*, *CG*, and *SBG* methods. However, when the number of tasks arrival increases further, the utility values are not increasing for MECs as all MECs are now at saturation point. Therefore when MECs reach the saturation point of resource availability, it is not possible to allocate more resources to demanding ESPs. Here we also can observe that MECs in the *NF* method have the lowest utility value than other methods. Also, the *SBG* and *CG* ensure very close utility values for the MECs.

In Fig. 3b, we observe that the average utility of ESPs also increases when the tasks arrival rate is increasing. The utility values are nearly similar for tasks arrival rate 200 to 400 for all described methods. The reason behind this is, at this tasks arrival rate, nearly all the demand of resources of all ESPs can be served in flexible manner as the resources are highly available due to low demand. However then the increasing rate of the utility of ESPs decreases when the tasks arrival rate is higher like 500 to 800. Also, when the tasks arrival rate is 700 - 800 as MECs are highly saturated therefore ESPs utility also reach a saturation level.

In Fig. 3c, we can see that increasing the rate of the arrival of the tasks decreases the average served percentage

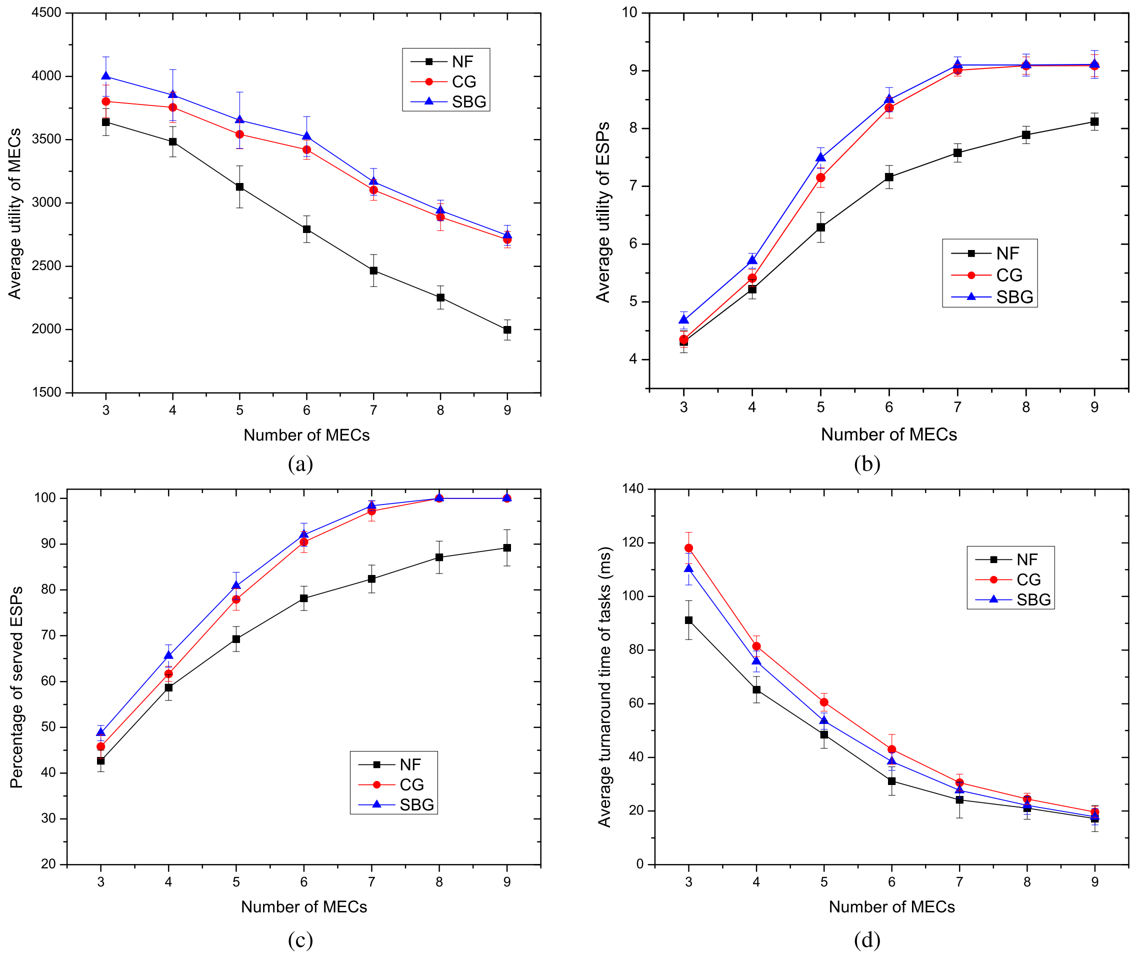

FIGURE 4. **Impact of varying the number of MECs on the performance of the studied systems.**

of ESPs. As already mentioned when more tasks are arriving, the demand of resources of ESPs also increases as they have to serve the increasing edge demand. However, the MECs are resource-constrained and all the MECs will face resource scarcity at the higher tasks arrival rate. For that particular reason more ESPs will face resource shortages and the number of ESPs getting service from MECs will drastically decrease. However as usual *NF* method performed worst in this case, *CG*, and our proposed method *SBG* performs moderately well to handle the increased resource demand.

In Fig. 3d, we observe that the *NF* method has the lowest TAT for successful tasks completion and *CG* has the highest. Our proposed *SBG* performs moderately than CG. Though NF has the lowest TAT it does not mean that *NF* is better than the other two methods as we already observed in Fig. 3c the percentage of served ESPs are very low. Both *CG* and *SBG* require extra communication time or burst time for searching and providing abundant resources of different MECs.

3) IMPACT OF VARYING THE NUMBER OF MECS

In figure 4 we change the number of MECs in the system from 3 to 9 and maintain the number of ESPs present in the system is 320. Also, we consider the tasks arrival rate at ESPs to 500 tasks per second and observe the impacts.

In Fig. 4a, increasing the number of MECs decreases the average utility value for all the MECs in all the described methods. When we increase the number of MECs it means we increase the total amount of resources to serve edge load. Now all the MEC servers will have fewer ESPs to serve, which means ESPs are now distributed between MECs more evenly. This will eventually decrease the utility of MECs. The *NF* method faces a drastic decrease in utility value, whereas the *CG* and our proposed *SBG* method behave more similarly when the number of MEC increases. We can see that when the number of MECs is 9 both *CG* and our proposed *SBG* have are nearly equal average utility for MECs because the need to spend less communication time and the demand of ESPs also nearly equally distributed among serving MECs.

In Fig. 4b, we observe that the average utility value of ESPs increases when the number of MECs increases. As already mentioned, we run this simulation by keeping the number of ESPs to 320, which is an overloaded situation for all MECs. That's why the utility value of ESPs is quite low when the MECs number is 3. After, when we increase the MEC count, which indicates we increase the resource availability. When the number of MECs is much higher like 7 to 9 we can see the gained utility of ESPs are nearly equal for both *CG* and *SBG* methods. This indicates that the system has more resources than the demand of ESPs and the utility value of ESPs is now at saturation point. However, the *NF* still performs poorly although we increase the number of MECs.

In Fig. 4c, we observe the percentage of served ESPs is increasing when the number of MECs increases. As already stated when the MECs number was low the scarcity of resources was very high. Which causes more than half of the ESPs not to get the required amount of CRBs. This situation eases when we increase resource availability by increasing the number of MECs in the system. All the three methods *NF*, *CG*, and *SBG* are follow the same intuition. We also observe that when the number of MECs is between 7 to 9, the percentage of served ESPs are close to 100% for both *CG* and our proposed *SBG* methods. That is all the ESPs are getting their required resources to fulfill the edge demand as the resource are abundant for all the MECs.

In Fig. 4d, we observe the decrease in average TAT concerning the increasing number of MECs. We see that the *NF* method has the lowest TAT value than the other two methods. The reason behind this is, in NF there is no possibility of renting resources from other MECs, thus this method serves its ESPs faster than rest two methods, although the served percentage is not satisfactory as we can see in Fig. 4c. However, our proposed *SBG* method has less TAT for successful tasks completion than *CG*. In this approach, the burst time or the communication time is lower than the *CG* method as finding resources is done by the MEC server itself which reduces the burst time. On the other hand in the *CG* method, MECs allocate resources to their own ESPs and then find other ESPs to provide abundant resources, this increases the burst time for tasks getting served. When the number of MEC servers reaches 8 to 9, we can also observe that all three methods have nearly equal average TAT for serving their ESPs.

VII. CONCLUSION

This paper developed a Stackelberg Game based dynamic resource allocation system that federates resources of lightly loaded MECs with the overloaded ones. The proposed distributed *SBG* system intelligently encashed the interactions among the buyer and seller MECs as well as those of an MEC with its ESPs. Therefore, it achieved optimal resource allocation while offering high level of fairness.

In future, state-of-the-art machine learning algorithms can be explored and exploited to estimate the resource requirements more accurately and thereby to increase the resource allocation efficiency further.

Special thanks to the ICT Division of the Government of Bangladesh for student fellowship.

REFERENCES

[1] V. Cisco. (2019). *Cisco Visual Networking Index: Forecast and Trends, 2017–2022 White Paper*. Accessed: Sep. 20, 2021. Porto Salvo, Lisboa. Disponível. [Online]. Available: https://www.cisco.com/c/pt_pt/about/press/news-archive-2018/20181127

[2] C. Campolo, A. Molinaro, A. Iera, and F. Menichella, "5G network slicing for vehicle-to-everything services," *IEEE Wireless Commun.*, vol. 24, no. 6, pp. 38–45, Dec. 2017.

[3] V. W. Wong, *Key Technologies for 5G Wireless Systems*. Cambridge, U.K.: Cambridge Univ. Press, 2017.

[4] Z. Chen, W. Hu, J. Wang, S. Zhao, B. Amos, G. Wu, K. Ha, K. Elgazzar, P. Pillai, R. Klatzky, D. Siewiorek, and M. Satyanarayanan, "An empirical study of latency in an emerging class of edge computing applications for wearable cognitive assistance," in *Proc. 2nd ACM/IEEE Symp. Edge Comput.*, Oct. 2017, pp. 1–14.

[5] T. Taleb, A. Ksentini, M. Chen, and R. Jantti, "Coping with emerging mobile social media applications through dynamic service function chaining," *IEEE Trans. Wireless Commun.*, vol. 15, no. 4, pp. 2859–2871, Apr. 2016.

[6] D. Tian, J. Zhou, Z. Sheng, M. Chen, Q. Ni, and V. C. M. Leung, "Self-organized relay selection for cooperative transmission in vehicular ad-hoc networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 10, pp. 9534–9549, Oct. 2017.

[7] T. X. Tran, A. Hajisami, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges," *IEEE Commun. Mag.*, vol. 55, no. 4, pp. 54–61, Apr. 2017.

[8] S. Kekki, W. Featherstone, Y. Fang, P. Kuure, A. Li, A. Ranjan, D. Purkayastha, F. Jiangping, D. Frydman, G. Verin, K.-W. Wen, K. Kim, R. Arora, A. Odgers, L. M. Contreras, and S. Scarpina, "MEC in 5G networks," *ETSI white paper*, vol. 28, pp. 1–28, Jun. 2018.

[9] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[10] T. G. Rodrigues, K. Suto, H. Nishiyama, and N. Kato, "Hybrid method for minimizing service delay in edge cloud computing through VM migration and transmission power control," *IEEE Trans. Comput.*, vol. 66, no. 5, pp. 810–819, May 1, 2017.

[11] J. Ahmed, M. M. Ashhab, M. A. Razzaque, and M. M. Rahman, "Execution delay-aware task assignment in mobile edge cloud and internet cloud," in *Proc. Int. Conf. Sustain. Technol. Ind. 4.0 (STI)*, Dec. 2019, pp. 1–6.

[12] Y. Zhang, X. Lan, Y. Li, L. Cai, and J. Pan, "Efficient computation resource management in mobile edge-cloud computing," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3455–3466, Dec. 2019.

[13] M. Liu, Y. Mao, S. Leng, and S. Mao, "Full-duplex aided user virtualization for mobile edge computing in 5G networks," *IEEE Access*, vol. 6, pp. 2996–3007, 2017.

[14] R. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw., Pract. Exper.*, vol. 41, no. 1, pp. 23–50, Aug. 2011.

[15] T. Kurze, M. Klems, D. Bermbach, A. Lenk, S. Tai, and M. Kunze, "Cloud federation," in *Proc. 2nd Int. Conf. Cloud Comput.*, Sep. 2011, pp. 32–38.

[16] A. Celesti, F. Tusa, M. Villari, and A. Puliafito, "How to enhance cloud architectures to enable cross-federation," in *Proc. IEEE 3rd Int. Conf. Cloud Comput.*, Jul. 2010, pp. 337–345.

[17] A. K. Das, T. Adhikary, M. A. Razzaque, E. J. Cho, and C. S. Hong, "A QoS and profit aware cloud confederation model for IaaS service providers," in *Proc. 8th Int. Conf. Ubiquitous Inf. Manage. Commun. (ICUIMC)*, 2014, pp. 1–7.

[18] Y. Li and M. Chen, "Software-defined network function virtualization: A survey," *IEEE Access*, vol. 3, pp. 2542–2553, 2015.

[19] D. Qiang, N. Ansari, and M. Toy, "Software-defined network virtualization: An architectural framework for integrating SDN and NFV for service provisioning in future networks," *IEEE Netw.*, vol. 30, no. 5, pp. 10–16, Sep./Oct. 2016.

[20] C. Gaul, M. Korner, and O. Kao, "Design and implementation of a cloud-federation agent for software defined networking," in *Proc. IEEE Int. Conf. Cloud Eng.*, Mar. 2015, pp. 323–328.

[21] I. Petri, M. Zou, A. R. Zamani, J. Diaz-Montes, O. Rana, and M. Parashar, "Integrating software defined networks within a cloud federation," in *Proc. 15th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, May 2015, pp. 179–188.

[22] M. Chen and Y. Hao, "Task offloading for mobile edge computing in software defined ultra-dense network," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 3, pp. 587–597, Mar. 2018.

[23] J. Xu, B. Palanisamy, H. Ludwig, and Q. Wang, "*Zenith*: Utility-aware resource allocation for edge computing," in *Proc. IEEE Int. Conf. Edge Comput. (EDGE)*, Jun. 2017, pp. 47–54.

[24] F. Zafari, K. K. Leung, D. Towsley, P. Basu, A. Swami, and J. Li, "Let's share: A game-theoretic framework for resource sharing in mobile edge clouds," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 2, pp. 2107–2122, Jun. 2021.
[25] Z. Liu, M. Lin, A. Wierman, S. H. Low, and L. L. Andrew, "Geographical load balancing with renewables," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 39, no. 3, pp. 62–66, Dec. 2011.
[26] H. Zhang, Y. Xiao, S. Bu, D. Niyato, R. Yu, and Z. Han, "Fog computing in multi-tier data center networks: A hierarchical game approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

JARGIS AHMED received the B.Sc. degree from the Department of Computer Science and Engineering, University of Dhaka, Bangladesh, in 2017, where he is currently pursuing the M.Sc. degree. His research interests include cloud computing, mobile edge computing, and the Internet of Things.

MD. ABDUR RAZZAQUE (Senior Member, IEEE) received the B.S. degree in applied physics and electronics and the M.S. degree in computer science from the University of Dhaka, Bangladesh, in 1997 and 1999, respectively, and the Ph.D. degree in computer engineering from Kyung Hee University, South Korea, in August 2009. From 2010 to 2011, he was a Research Professor with the College of Electronics and Information, Kyung Hee University, South Korea. From 2016 to 2021, he worked with the Green University of Bangladesh at different periods as a Pro Vice Chancellor, the Dean of the Faculty of Science and Engineering, and the Chairperson of the Department of CSE. He worked as a Visiting Professor with Stratford University, Falls Church, VA, USA, in 2017. He is currently a Professor with the Department of Computer Science and Engineering, University of Dhaka. He is the Director of the Green Networking Research Group (http://cse.du.ac.bd/gnr), Department of CSE, DU. He has published more than 140 research papers in international conferences and journals. His research interests include modeling, analysis, and optimization of wireless networking protocols and architectures, mobile crowdsourcing, sensor data clouds, the Internet of Things, and edge computing. He is a TPC Member of IEEE HPCC, ICOIN, SCALCOM, SKIMA, ICIEV, ADM, NSysS, and ICACCI. He is a member of the IEEE Computer Society. He is the General Chair of STI 2021–2019 and the TPC Chair of ICIET 2019–2018. He is chairing 2021 Executive Committee for IEEE Computer Society Bangladesh Chapter. He is an Associate Editor of IEEE ACCESS. He is an Editorial Board Member of the *Journal of Networks and Applications*.

MD. MUSTAFIZUR RAHMAN (Member, IEEE) received the B.Sc. degree (Hons.) in applied physics and electronics and the M.Sc. degree in computer science from the University of Dhaka, Bangladesh, in 1994 and 1997, respectively, and the Ph.D. degree in computer engineering from Kyung Hee University, South Korea, in 2010. He is currently a Professor with the Department of Computer Science and Engineering, University of Dhaka. His research interest includes multiple access protocols.

SALMAN A. ALQAHTANI (Member, IEEE) is currently a Professor with the Department of Computer Engineering, King Saud University, Riyadh. His current research interests include 5G networks, broadband wireless communications, radio resource management for 4G and beyond networks (call admission control, packet scheduling, and radio resource sharing techniques), cognitive and cooperative wireless networking, small cell and heterogeneous networks, self-organizing networks, SDN/NFV, 5G network slicing, smart grid, the intelligent IoT solutions for smart cities, dynamic spectrum access, co-existence issues on heterogeneous networks in 5G, industry 4.0 issues, the Internet of Everything, mobile edge and fog computing, cyber sovereignty, performance evaluation and analysis of high-speed packet switched networks, system model and simulations, and integration of heterogeneous wireless networks. Mainly his focus is on the design and optimization of 5G MAC layers, closed-form mathematical performance analysis, energy-efficiency, and resource allocation and sharing strategies. He has authored two scientific books and authored/coauthored around 76 journals and conference papers in the topic of his research interests, since 2004. He serves as a reviewer for several national and international journals.

MOHAMMAD MEHEDI HASSAN (Senior Member, IEEE) received the Ph.D. degree in computer engineering from Kyung Hee University, South Korea, in February 2011. He is currently a Full Professor with the Department of Information Systems, College of Computer and Information Sciences (CCIS), King Saud University (KSU), Riyadh, Saudi Arabia. He has authored or coauthored around more than 180 publications, including refereed IEEE/ACM/Springer/Elsevier journals, conference papers, books, and book chapters. Recently, his four publications have been recognized as the ESI Highly Cited Papers. His research interests include cloud computing, edge computing, the Internet of Things, body sensor networks, big data, deep learning, mobile cloud, smart computing, wireless sensor networks, 5G networks, and social networks. He has served as the chair and a technical program committee member for international conferences.